

# Alpha PCI 164-266/300

---

## Design Guide

Order Number: EK-AL266-UG. A01

This manual describes the Alpha PCI 164-266 (order number EBP40-AN) and Alpha PCI 164-300 (order number EBP41-AN) motherboards configured for the Microsoft Windows NT operating system.

**Revision/Update Information:** This is a new manual.

**Digital Equipment Corporation  
Maynard, Massachusetts**

---

**April 1996**

Possession, use, or copying of the software described in this publication is authorized only pursuant to a valid written license from Digital or an authorized sublicensor.

While Digital believes the information included in this publication is correct as of the date of publication, it is subject to change without notice.

Digital Equipment Corporation makes no representations that the use of its products in the manner described in this publication will not infringe on existing or future patent rights, nor do the descriptions contained in this publication imply the granting of licenses to make, use, or sell equipment or software in accordance with the description.

© Digital Equipment Corporation 1995, 1996.

All rights reserved.  
Printed in U.S.A.

AlphaGeneration, DEC, DECchip, DECladebug, Digital, Digital Semiconductor, VAX DOCUMENT, the AlphaGeneration design mark, and the DIGITAL logo are trademarks of Digital Equipment Corporation.

Digital Semiconductor is a Digital Equipment Corporation business.  
Digital UNIX Version 3.2 for Alpha is a UNIX 93 branded product.

Hewlett-Packard is a registered trademark of Hewlett-Packard Company.  
IEEE is a registered trademark of The Institute of Electrical and Electronics Engineers, Inc.  
Intel and UPI are trademarks of Intel Corporation.  
Motorola is a registered trademark of Motorola, Inc.  
National is a registered trademark of National Semiconductor Corporation.  
NT and Windows NT are trademarks of Microsoft Corporation.  
Powerview is a trademark of Viewlogic Systems, Inc.  
Prentice Hall is a registered trademark of Prentice-Hall, Inc. of Englewood Cliffs, NJ.  
TI is a registered trademark of Texas Instruments Inc.  
TriQuint is a registered trademark of TriQuint Semiconductor, Inc.  
UNIX is a registered trademark in the United States and other countries, licensed exclusively through X/Open Company Ltd.  
VxWorks is a registered trademark of Wind River Systems, Inc.  
Xilinx is a trademark of Xilinx Incorporated.

All other trademarks and registered trademarks are the property of their respective holders.

This document was prepared using VAX DOCUMENT, Version 2.1.

---

# Contents

<b>About This Guide</b> .....	ix
<b>1 Introduction to the Alpha PCI 164-266/300</b>	
1.1 Board Components and Features .....	1-1
1.1.1 DECchip 21171 Core Logic Chipset .....	1-3
1.1.2 Memory Subsystem .....	1-3
1.1.3 L3 Bcache Subsystem Overview .....	1-3
1.1.4 PCI Interface Overview .....	1-4
1.1.5 ISA Interface Overview .....	1-4
1.1.6 Miscellaneous Logic .....	1-4
1.1.7 Software Support .....	1-5
1.2 Board Uses .....	1-6
<b>2 Board Configuration and Connectors</b>	
2.1 Configuration Jumpers .....	2-1
2.2 Alpha PCI 164-266/300 Connectors .....	2-8
<b>3 Functional Description</b>	
3.1 Alpha PCI 164-266/300 Bcache Interface .....	3-2
3.2 DECchip 21171 Chipset .....	3-4
3.2.1 CIA Chip Overview .....	3-4
3.2.2 DSW Overview .....	3-6
3.3 Main Memory Interface .....	3-6
3.4 PCI Devices .....	3-7
3.4.1 Saturn-IO (SIO) Chip .....	3-8
3.4.2 PCI Expansion Slots .....	3-8
3.5 ISA Bus Devices .....	3-9
3.5.1 Combination Controller .....	3-9
3.5.2 Keyboard and Mouse Controller .....	3-12
3.5.3 Time-of-Year Clock .....	3-12

3.5.4	Utility Bus Memory Device .....	3-13
3.5.5	ISA Expansion Slots .....	3-13
3.6	Interrupts .....	3-14
3.7	System Clocks .....	3-17
3.8	Reset and Initialization .....	3-20
3.9	Serial ROM .....	3-22
3.10	dc Power Distribution .....	3-23
3.11	System Software .....	3-25
3.11.1	Serial ROM Code .....	3-25
3.11.2	Mini-Debugger Code .....	3-25
3.11.3	Debug Monitor ROM Code .....	3-26
3.11.4	Operating Systems .....	3-26

## 4 System Address Mapping

4.1	Physical Memory Regions .....	4-1
4.2	Alpha 21164 Address Mapping to PCI Space .....	4-4
4.2.1	Cacheable Memory Space (00.0000.0000 Through 00.3FFF.FFFF) .....	4-5
4.2.2	PCI Sparse Memory Space (80.0000.0000 Through 85.7FFF.FFFF) .....	4-5
4.2.3	Hardware Extension Registers (HAE) .....	4-9
4.2.4	PCI Sparse I/O Space (85.8000.0000 Through 85.FFFF.FFFF) .....	4-10
4.2.5	PCI Dense Memory Space (86.0000.0000 Through 86.FFFF.FFFF) .....	4-13
4.2.6	PCI Configuration Space (87.0000.0000 Through 87.1FFF.FFFF) .....	4-14
4.2.7	PCI Interrupt Acknowledge/Special Cycle Space (87.2000.0000 Through 87.3FFF.FFFF) .....	4-19
4.2.8	Alpha PCI 164-266/300 Hardware-Specific and Miscellaneous Register Space (87.4000.0000 Through 87.6FFF.FFFF) .....	4-19
4.2.9	PCI-to-Physical Memory Addressing .....	4-20

## 5 Power and Environmental Requirements

5.1	Power Requirements .....	5-1
5.2	Environmental Requirements .....	5-2
5.3	Physical Board Parameters .....	5-2

## A I/O Space Address Maps

A.1	PCI Sparse Memory Space . . . . .	A-1
A.2	PCI Sparse I/O Space . . . . .	A-1
A.2.1	PCI Sparse I/O Space—Region A . . . . .	A-2
A.2.1.1	PC87312 Combination Controller Register Address Space . . . . .	A-2
A.2.1.2	8242AH Keyboard and Mouse Controller Addresses . . . . .	A-6
A.2.1.3	Time-of-Year (TOY) Clock Addresses . . . . .	A-6
A.2.1.4	Flash ROM Segment Select Register . . . . .	A-7
A.2.1.5	Configuration Jumpers (CONF4—CONF15) . . . . .	A-8
A.2.1.6	Interrupt Control PLD Addresses . . . . .	A-8
A.2.2	PCI Sparse I/O Space—Region B . . . . .	A-8
A.3	PCI Dense Memory Space . . . . .	A-13
A.3.1	Flash ROM Memory Addresses . . . . .	A-13
A.3.2	Map of Flash ROM Memory . . . . .	A-13
A.3.3	Flash ROM Configuration Registers . . . . .	A-14
A.4	PCI Configuration Address Space . . . . .	A-16
A.4.1	SIO PCI-to-ISA Bridge Configuration Address Space . . . . .	A-16
A.5	PCI Interrupt Acknowledge/Special Cycle Address Space . . . . .	A-18
A.6	Hardware-Specific and Miscellaneous Register Space . . . . .	A-18
A.6.1	CIA Main CSR Space . . . . .	A-18
A.6.2	CIA Memory Control CSR Space . . . . .	A-19
A.6.3	CIA PCI Address Translation Map Space . . . . .	A-20
A.7	Alpha 21164 Microprocessor Cbox IPR Space . . . . .	A-22

## B SRAM Initialization

B.1	SRAM Initialization . . . . .	B-1
B.2	Firmware Interface . . . . .	B-2
B.3	Automatic CPU Speed Detection . . . . .	B-4
B.4	CPU Bus Interface Timing . . . . .	B-4
B.5	Bcache Read and Write Timing Calculations . . . . .	B-7
B.5.1	Read Cycle Calculation . . . . .	B-7
B.5.2	Write Cycle Calculations . . . . .	B-7
B.5.3	Read/Write Spacing Calculations . . . . .	B-9
B.6	Memory Initialization . . . . .	B-9
B.7	Bcache Initialization . . . . .	B-9
B.8	Special ROM Header . . . . .	B-10
B.9	Flash ROM Structure . . . . .	B-13
B.10	Flash ROM Access . . . . .	B-16
B.11	Icache Flush Code . . . . .	B-17

## C Technical Support and Additional Documentation

C.1	Technical Support . . . . .	C-1
C.2	Ordering Associated Literature . . . . .	C-2
C.3	Ordering Third-Party Literature . . . . .	C-3

## Glossary

## Index

## Figures

1-1	Alpha PCI 164-266/300 Functional Block Diagram . . . . .	1-2
2-1	Alpha PCI 164-266/300 Jumper Locations . . . . .	2-2
2-2	Configuration Jumpers . . . . .	2-3
2-3	Alpha PCI 164-266/300 Connector Locations . . . . .	2-9
2-4	Detail of Header Connector J2 . . . . .	2-10
3-1	Alpha PCI 164-266/300 Bcache Array . . . . .	3-2
3-2	Main Memory Interface . . . . .	3-5
3-3	ISA Devices . . . . .	3-10
3-4	Interrupt Logic . . . . .	3-15
3-5	System Clocks and Distribution . . . . .	3-18
3-6	System Reset and Initialization . . . . .	3-21
3-7	SRROM and Serial Port . . . . .	3-23
3-8	dc Power Distribution . . . . .	3-24
4-1	PCI Sparse Memory Space Address Translation—Region 1 . . . . .	4-8
4-2	PCI Sparse Memory Space Address Translation—Region 2 . . . . .	4-9
4-3	PCI Sparse Memory Space Address Translation—Region 3 . . . . .	4-9
4-4	PCI Sparse I/O Space Address Translation—Region A . . . . .	4-12
4-5	PCI Sparse I/O Space Address Translation—Region B . . . . .	4-12
4-6	Addressing Diagram: PCI Target Window Compare . . . . .	4-22
4-7	PCI to System Bus Scatter-Gather Address Translation Map . . . . .	4-25
5-1	Board Component Layout . . . . .	5-3
B-1	Write Cycle Timing . . . . .	B-7

B-2	Special Header Content .....	B-10
-----	------------------------------	------

**Tables**

1-1	Main Memory Sizes .....	1-3
2-1	Configuration Jumper Position Descriptions .....	2-4
2-2	Alpha PCI 164-266/300 Connector Descriptions .....	2-11
2-3	List of Connector and Jumper Part Numbers .....	2-16
3-1	Bcache Configurations .....	3-3
3-2	Alpha PCI 164-266/300 System Interrupts .....	3-14
3-3	PCI-to-ISA SIO Bridge Interrupts .....	3-16
4-1	Three Physical Memory Regions .....	4-2
4-2	Physical Memory Regions (Detailed) .....	4-3
4-3	PCI Sparse Memory Space Byte-Enable Generation .....	4-7
4-4	INT4_VALID to Address Translation for Sparse Write Operations .....	4-7
4-5	High-Order Sparse Space Bits .....	4-8
4-6	PCI Sparse I/O Space Byte-Enable Generation .....	4-11
4-7	PCI Configuration Space Definition .....	4-16
4-8	PCI Configuration Space Byte-Enable Generation .....	4-17
4-9	CPU Address Encoding for PCI Device Selection .....	4-18
4-10	Alpha PCI 164-266/300 Primary PCI IDSEL Mapping .....	4-19
4-11	Hardware-Specific Register Space .....	4-20
4-12	PCI Target Window Enables .....	4-21
4-13	PCI Target Address Translation—Direct Mapped (SG Mapping Disabled) .....	4-23
4-14	Scatter-Gather Map Address .....	4-24
5-1	Power Supply dc Current Requirements .....	5-1
5-2	Board Component List .....	5-4
A-1	PC87312 Combination Controller Register Address Space Map .....	A-2
A-2	Keyboard and Mouse Controller Addresses .....	A-6
A-3	Time-of-Year Clock Device Addresses .....	A-7
A-4	Flash ROM Segment Select Register .....	A-7
A-5	Configuration Jumpers (CONF4—CONF15) .....	A-8
A-6	Interrupt Control PLD Addresses .....	A-8
A-7	SIO PCI-to-ISA Bridge Operating Register Address Space Map .....	A-9

A-8	Flash ROM Memory Addresses (Within Segment) . . . . .	A-13
A-9	Map of Flash ROM Memory . . . . .	A-13
A-10	Flash ROM Configuration Registers . . . . .	A-15
A-11	Address Bits and PCI Device IDSEL Pins . . . . .	A-16
A-12	SIO PCI-to-ISA Bridge Configuration Address Space Map . . . . .	A-17
A-13	CIA Control, Diagnostic, and Error Registers . . . . .	A-18
A-14	CIA Memory Control Registers . . . . .	A-19
A-15	PCI Address Translation Registers . . . . .	A-20
A-16	Alpha 21164 Cache Configuration Registers . . . . .	A-22
B-1	Output Parameter Descriptions . . . . .	B-2
B-2	Cache Loop Delay Characteristics . . . . .	B-5
B-3	Typical SRAM Specifications . . . . .	B-5
B-4	CPU Specifications . . . . .	B-5
B-5	Special Header Entry Descriptions . . . . .	B-11
B-6	Flash ROM Image Selection . . . . .	B-13



---

## About This Guide

This guide describes Digital's Alpha PCI 164-266/300 motherboard based on the 21164 microprocessor and DECchip 21171 chipset.

### Audience

This guide is intended for system designers and others who are intending to embed a compute engine into their system. It is written to assist the designer in familiarization with the predominant features of Digital's real-time motherboards and Digital's Alpha family of microprocessors.

### Scope

This guide describes the features, configuration, functional operation, and interfaces of the Alpha PCI 164-266/300 motherboard. This guide does not include specific bus specifications (for example, PCI or ISA buses).

Additional information is available from vendor documentation and IEEE specifications. See Appendix C for information about how to order related documentation and obtain additional technical support.

### Content

This guide contains the following chapters and appendixes:

- Chapter 1, Introduction to the Alpha PCI 164-266/300, is an overview of the Alpha PCI 164-266/300, including its components, uses, and features.
- Chapter 2, Board Configuration and Connectors, describes the user environment configuration; board connectors and functions; jumper functions; and identifies jumper and connector locations.
- Chapter 3, Functional Description, provides a functional description of the board, including the DECchip 21171 chipset, L3 backup cache (Bcache) and memory subsystems, interrupts, clock and power subsystems, and peripheral component interconnect (PCI) and Industry Standard Architecture (ISA) devices.

- Chapter 4, System Address Mapping, describes the mapping of the 40-bit processor address space into memory and I/O space addresses.
- Chapter 5, Power and Environmental Requirements, describes the board power and environmental requirements, and identifies major board components.
- Appendix A, I/O Address Maps, lists the physical Alpha PCI 164-266/300 PCI address spaces and regions, including control, I/O interface, and address operating registers and PCI/ISA device registers.
- Appendix B, SROM Initialization, describes the general serial read-only memory (SROM), Bcache, and memory initialization steps and associated parameters. Also included are the firmware interface, timing considerations, and SROM header information.
- Appendix C, Technical Support and Additional Documentation, lists other relevant Digital documentation and gives addresses for ordering documentation from other vendors.
- Glossary lists and defines terms associated with the Alpha PCI 164-266 /300.

## Document Conventions

This section provides the conventions used in this document.

### Bit and Field Abbreviations

The following list describes the bit and field abbreviations:

Bit/Field Abbreviation	Description
RO (read only)	Bits and fields specified as RO can be read but not written.
RW (read/write)	Bits and fields specified as RW can be read and written.
WO (write only)	Bits and fields specified as WO can be written but not read.

### Bit Notation

Multiple bit fields are shown as extents (see Ranges and Extents in this section).

### Caution

Cautions indicate potential damage to equipment or data.

## Data Units

The following data unit terminology, common within Digital, is used throughout this guide:

Term	Words	Bytes	Bits
Word	1	2	16
Longword	2	4	32
Quadword	4	8	64
Octaword	8	16	128
Hexword	16	32	256

### Note

Notes provide additional information.

### Numbering

All numbers are decimal or hexadecimal unless otherwise indicated. In case of ambiguity, a subscript indicates the radix of nondecimal numbers. For example, 19 is a decimal number, but  $19_{16}$  and 19A are hexadecimal numbers.

### UNPREDICTABLE and UNDEFINED

Throughout this manual, the terms UNPREDICTABLE and UNDEFINED are used. Their meanings are quite different and must be carefully distinguished.

In particular, only privileged software (that is, software running in kernel mode) can trigger UNDEFINED operations. Unprivileged software cannot trigger UNDEFINED operations. However, either privileged or unprivileged software can trigger UNPREDICTABLE results or occurrences.

UNPREDICTABLE results or occurrences do not disrupt the basic operation of the processor. The processor continues to execute instructions in its normal manner. In contrast, UNDEFINED operations can halt the processor or cause it to lose information.

The terms UNPREDICTABLE and UNDEFINED can be further described as follows:

- **UNPREDICTABLE**
  - Results or occurrences specified as UNPREDICTABLE might vary from moment to moment, implementation to implementation, and instruction to instruction within implementations. Software can never depend on results specified as UNPREDICTABLE.

- An UNPREDICTABLE result might acquire an arbitrary value subject to a few constraints. Such a result might be an arbitrary function of the input operands or of any state information that is accessible to the process in its current access mode. UNPREDICTABLE results may be unchanged from their previous values.

Operations that produce UNPREDICTABLE results might also produce exceptions.

- An occurrence specified as UNPREDICTABLE might happen or not based on an arbitrary choice function. The choice function is subject to the same constraints as are UNPREDICTABLE results and, in particular, must not constitute a security hole.

Specifically, UNPREDICTABLE results must not depend upon, or be a function of the contents of memory locations or registers that are inaccessible to the current process in the current access mode.

Also, operations that might produce UNPREDICTABLE results must not:

- Write or modify the contents of memory locations or registers to which the current process in the current access mode does not have access.
- Halt or hang the system or any of its components.

For example, a security hole would exist if some UNPREDICTABLE result depended on the value of a register in another process, on the contents of processor temporary registers left behind by some previously running process, or on a sequence of actions of different processes.

- **UNDEFINED**

- Operations specified as UNDEFINED can vary from moment to moment, implementation to implementation, and instruction to instruction within implementations. The operation can vary in effect from nothing, to stopping system operation.
- UNDEFINED operations can halt the processor or cause it to lose information. However, UNDEFINED operations must not cause the processor to hang, that is, reach an unhalted state from which there is no transition to a normal state in which the machine executes instructions. Only privileged software (that is, software running in kernel mode) can trigger UNDEFINED operations.

### **Data Field Size**

The term  $INT_{nn}$ , where  $nn$  is one of 2, 4, 8, 16, 32, or 64, refers to a data field of  $nn$  contiguous NATURALLY ALIGNED bytes. For example,  $INT_4$  refers to a NATURALLY ALIGNED longword.

### **Ranges and Extents**

Ranges are specified by a pair of numbers separated by two periods (..) and are inclusive. For example, a range of 0..4 includes the integers 0, 1, 2, 3, and 4.

Extents are specified by a single number, or a pair of numbers in angle brackets (< >) separated by a colon (:). For example, bits <7:3> specify an extent including bits 7, 6, 5, 4, and 3.

### **Register and Memory Figures**

Register figures have bit and field position numbering starting at the right (low-order) and increasing to the left (high-order).

Memory figures have addresses starting at the top and increasing toward the bottom.

### **Signal Names**

Signal names in text are printed in boldface lowercase type. For example, “. . . bits **data<127:0>** are delivered to the Bcache SIMM connectors . . . ”



---

# Introduction to the Alpha PCI 164-266/300

This chapter provides an overview of the Alpha PCI 164-266/300, its components, features, and uses. The Alpha PCI 164-266/300 is based on the Alpha 21164 Microprocessor and the 21171 support chip set.

The Alpha PCI 164-266/300 can be used as a single-board hardware and software development platform for the design, integration, and analysis of supporting logic and subsystems. The board also provides a platform for PCI I/O device hardware and software development.

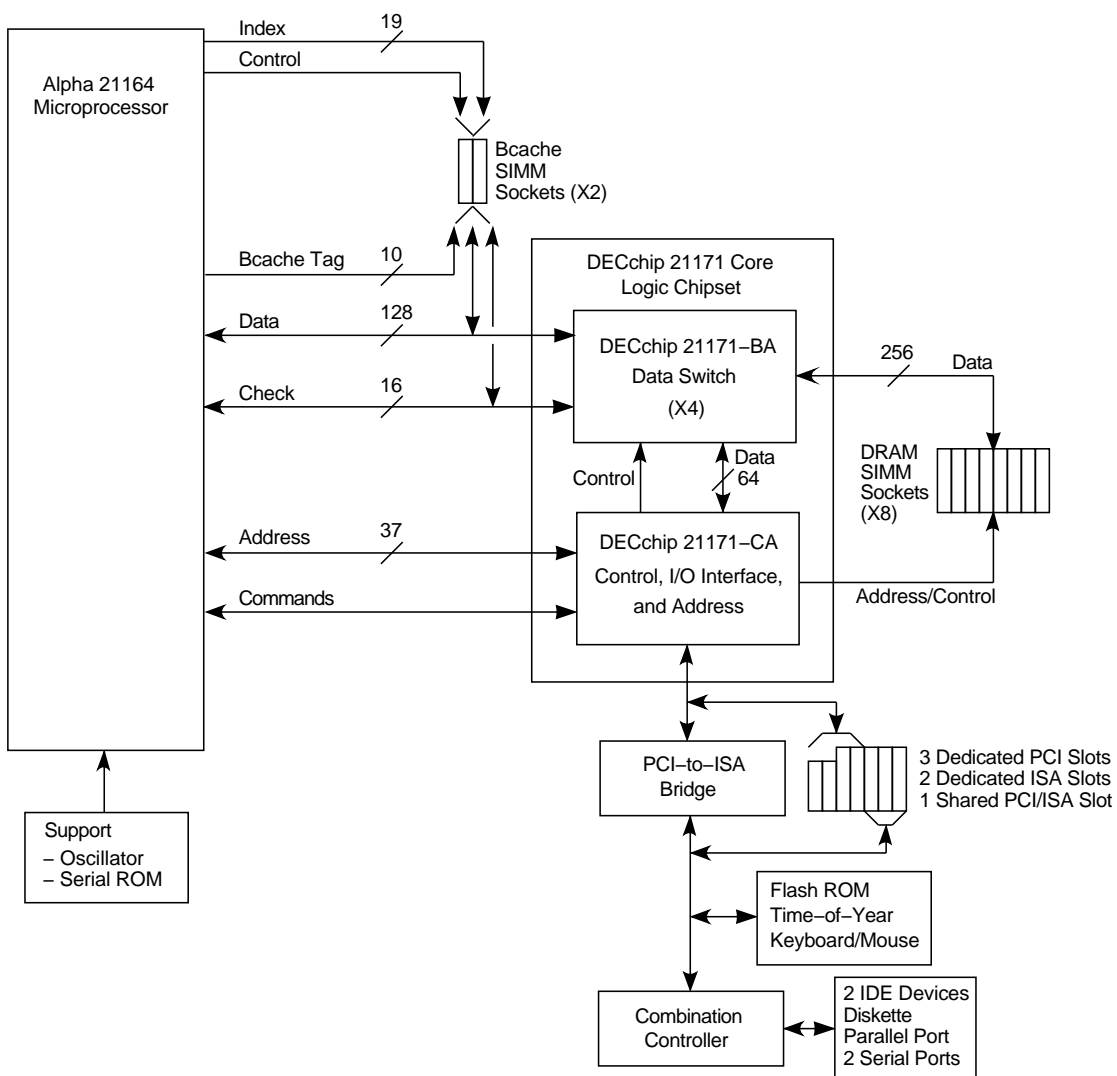
Appendix C provides ordering information and a list of related documentation.

## 1.1 Board Components and Features

The Alpha PCI 164-266/300 is implemented in industry-standard parts and uses an Alpha 21164 CPU running at 266 MHz or 300 MHz, depending on the model number. The functional components are shown in Figure 1-1 and are introduced in the following subsections.

## 1.1 Board Components and Features

Figure 1-1 Alpha PCI 164-266/300 Functional Block Diagram





## 1.1 Board Components and Features

### 1.1.1 DECchip 21171 Core Logic Chipset

The 21164 is supported by the DECchip 21171 chipset. The chipset consists of the following two application-specific integrated circuit (ASIC) types:

- One copy of the DECchip 21171-CA control, I/O interface, and address chip (CIA) provides the interface between the 21164, main memory (addressing and control), and the peripheral component interconnect (PCI) bus. It also provides the data switch companion chips with control information to direct the data flow.
- Four copies of the DECchip 21171-BA data switch chip (DSW) provide the memory interface data path and route PCI data through the CIA chip.

The chipset includes the majority of functions required to develop a high-performance PC or workstation, requiring minimum discrete logic on the module. It provides flexible and generic functions to allow its use in a wide range of systems.

### 1.1.2 Memory Subsystem

The dynamic random-access memory (DRAM) provides 32MB to 512MB with a 256-bit data bus. The memory is contained in one bank of eight commodity single inline memory modules (SIMMs). Single- or double-sided SIMMs may be used. Each SIMM is 36 bits wide, with 32 data bits and 4 check bits, with 70 ns or less access. Table 1–1 lists the SIMM sizes supported and the corresponding main memory size for 256-bit arrays.

**Table 1–1 Main Memory Sizes**

<b>SIMM Size</b>	<b>Eight SIMMs (256-Bit Array)</b>
1M × 36	32MB
2M × 36	64MB
4M × 36	128MB
8M × 36	256MB
16M × 36	512MB

### 1.1.3 L3 Bcache Subsystem Overview

The Alpha PCI 164-266/300 is shipped with an L3 backup cache of 2MB, but the board-level external L3 backup cache (Bcache) subsystem supports multiple cache sizes and access times. Cache sizes supported are 2MB with Alpha cache SIMMs populated with 128K × 8 SRAMs, and 4MB and 8MB with SIMMs populated with 512K × 8 SRAMs. Speeds of 6 ns to 15 ns can be used. Wave

## 1.1 Board Components and Features

pipelining decreases the cache loop times by one CPU cycle in most cases. The Bcache size can be reconfigured through onboard hardware jumpers. As implemented in the Alpha PCI 164-266/300, the Bcache operates in 64-byte mode only.

### 1.1.4 PCI Interface Overview

The Alpha PCI 164-266/300 PCI interface is the main I/O bus for the majority of functions (SCSI interface, graphics accelerator, and so on). The PCI interface provides a selectable PCI speed between 25 MHz and 33 MHz (based on the 21164 clock divisor). An onboard PCI-to-ISA bridge is provided through an Intel 82378ZB Saturn-I/O chip (SIO).

The PCI bus has three dedicated PCI expansion slots (one 64-bit and two 32-bit) and one shared 64-bit PCI/ISA slot.

### 1.1.5 ISA Interface Overview

The ISA bus has two dedicated slots and a third shared ISA/PCI slot. It provides the following system support functions:

- Mouse and keyboard controller functions—provided by an Intel 8242 chip.
- An IDE interface, a diskette controller, two universal asynchronous receiver-transmitters (UARTs) with full modem control, and a bidirectional parallel port—provided by a National 87312 combination chip.
- A time-of-year (TOY) function—provided by a Dallas Semiconductor DS1287 chip.
- Operating system support—provided by a 1MB flash ROM that contains firmware and debug monitor code.

Users can develop code on a host system, and load software into the Alpha PCI 164-266/300 through a serial line, diskette, or Ethernet board. In addition, sectors of the flash ROM can be programmed for application-specific purposes.

### 1.1.6 Miscellaneous Logic

The Alpha PCI 164-266/300 contains the following miscellaneous components:

- Clocks  
A 26.66-MHz oscillator and phase-locked loop (PLL) clock generator provide a clock source to the 21164 microprocessor and system.  
A 14.3-MHz crystal and frequency generator provide a clock source for ISA devices.
- Serial ROM

## 1.1 Board Components and Features

A Xilinx XC17128 serial ROM (SROM) contains initial code that is loaded into the 21164 instruction cache (Icache) on power-up. A serial line interface is also provided to allow direct connection to a terminal line for debugging purposes.

- Programmable array logic (PAL) devices for the following functions:
  - One PAL for utility bus (Ubus) decoding
  - One PAL for interrupts
  - Two PAL devices for memory row address strobe (RAS) bank generation and buffering

### 1.1.7 Software Support

Software support code, consisting of a debug monitor and Windows NT ARC firmware is contained in a 1MB flash ROM. Development code can be generated on a host system and loaded into the Alpha PCI 164-266/300 through a serial line, an Ethernet board, or a diskette. In addition, sectors of the flash ROM can be programmed for an application-specific purpose. The debug monitor provides functions that let you:

- Download files through serial and Ethernet ports and diskette.
- Load data from a ROM through the debug monitor.
- Examine and deposit the Alpha PCI 164-266/300 system registers, a few 21164 internal processor registers (IPRs), and I/O mapped registers.
- Examine and modify DRAM and I/O mapped memory.
- Disassemble CPU instructions in memory.
- Transfer control to programs in memory.
- Perform native debugging operations, including breakpoints and single stepping.
- Perform full source-level debugging operations by using DECladebug software running on a host communicating through an Ethernet connection.
- Perform a memory image dump.

## 1.2 Board Uses

### 1.2 Board Uses

The Alpha PCI 164-266/300 has a remote debugging capability and a software debug monitor for loading code into the board and for performing other software debugging functions such as memory read, memory write, and instruction breakpoint.

When combined with a hardware interface, the debug monitor can be used to write and debug software (for example, device drivers) for workstation and PC-type products.

The Alpha PCI 164-266/300 provides a basis for a high-performance, low-cost desktside PC or workstation.

---

## Board Configuration and Connectors

The Alpha PCI 164-266/300 uses jumpers to implement configuration parameters such as variations in backup cache (Bcache) size, access timing, and speed, as well as boot parameters. These jumpers must be configured for the user's environment. Onboard connectors are provided for the I/O interfaces, single inline memory modules (SIMMs), and serial and parallel peripheral ports.

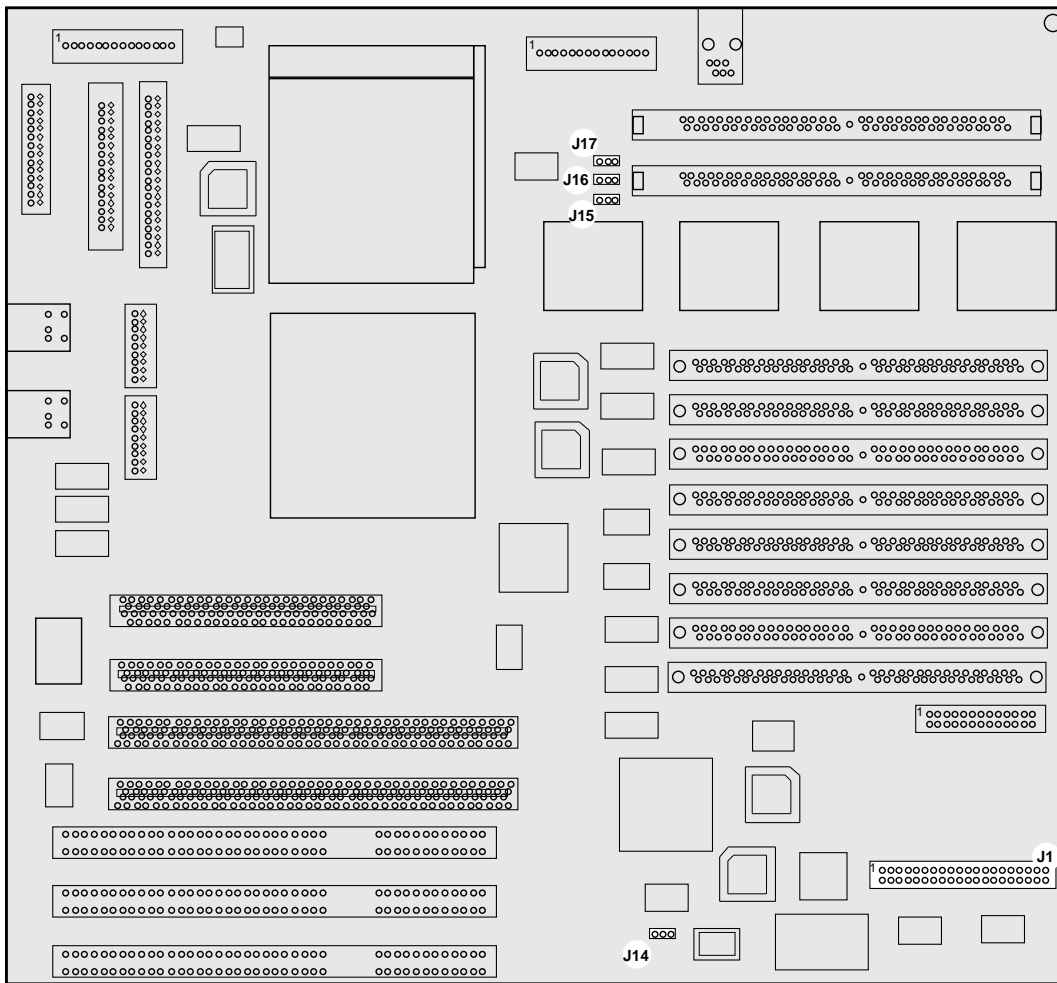
After the module is configured, power can be applied, and the debug monitor can be run. The debug monitor and its commands are described in the *Alpha Microprocessors Motherboard Debug Monitor User's Guide*.

### 2.1 Configuration Jumpers

Figure 2-1 identifies the location of the software and hardware configuration jumpers, and Table 2-1 provides descriptions. Figure 2-2 provides a detailed view of the configuration jumpers and their function.

## 2.1 Configuration Jumpers

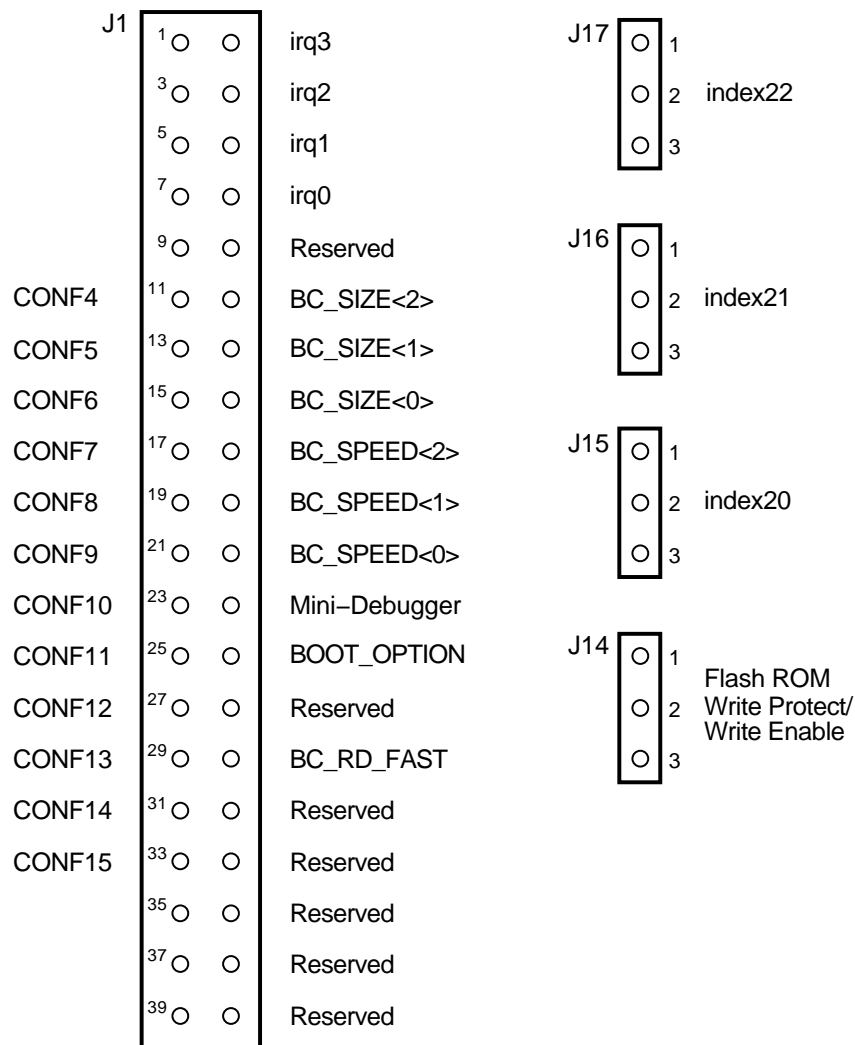
Figure 2–1 Alpha PCI 164-266/300 Jumper Locations



MK-2306-30

## 2.1 Configuration Jumpers

Figure 2–2 Configuration Jumpers



MK-2306-04

## 2.1 Configuration Jumpers

**Table 2–1 Configuration Jumper Position Descriptions**

Feature	Jack/Jumper—Pins and Description				
System clock divisor	J1—1/2, —3/4, —5/6, —7/8				
	<b>J1—1/2 (irq3)</b>	<b>J1—3/4 (irq2)</b>	<b>J1—5/6 (irq1)</b>	<b>J1—7/8 (irq0)</b>	<b>Ratio<sup>1</sup></b>
	In	In	Out	Out	3
	In	Out	In	In	4
	In	Out	In	Out	5
	In	Out	Out	In	6
	In	Out	Out	Out	7
	Out	In	In	In	8
	<b>Divisor 8 is used for 266 MHz (default).</b>				
	Out	In	In	Out	9
	<b>Divisor 9 is used for 300 MHz.</b>				
	Out	In	Out	In	10
	Out	In	Out	Out	11
	Out	Out	In	In	12
	Out	Out	In	Out	13
Out	Out	Out	In	14	
Out	Out	Out	Out	15	

<sup>1</sup>The Alpha PCI 164-266/300 supports PCI bus speeds of 25 MHz to 33 MHz. Not all system clock ratios are permissible at each CPU speed. The following CPU speed/ratio combinations are

allowed:	<b>CPU Speed</b>	<b>Ratios Allowed</b>
	266.6 MHz	8, 9, and 10
	300 MHz	

(continued on next page)



## 2.1 Configuration Jumpers

**Table 2–1 (Cont.) Configuration Jumper Position Descriptions**

Feature	Jack/Jumper—Pins and Description			
BC_SIZE<2:0>	J1—11/12 (CONF4), —13/14 (CONF5), —15/16 (CONF6)			
	These jumpers allow the Bcache to emulate the sizes specified in the following table. These jumpers are changed in conjunction with the appropriate index jumpers J17, J16, and J15.			
	<b>CONF4 Pins 11/12</b>	<b>CONF5 Pins 13/14</b>	<b>CONF6 Pins 15/16</b>	<b>Bcache</b>
	In	In	In	Reserved
	In	In	Out	Reserved
	In	Out	In	Reserved
	In	Out	Out	2MB (default)
	Out	In	In	4MB
	Out	In	Out	8MB
	Out	Out	In	Reserved
	Out	Out	Out	Reserved
Bcache size— index address bits <22:20>	J17, J16, J15			
	<b>Jumper</b>	<b>2MB<sup>2</sup> (default)</b>	<b>4MB<sup>3</sup></b>	<b>8MB<sup>3</sup></b>
	J17 (index22)	2 to 3	2 to 3	1 to 2
	J16 (index21)	2 to 3	1 to 2	1 to 2
	J15 (index20)	1 to 2	1 to 2	1 to 2

<sup>2</sup>SIMMs populated with 128K × 8 or 512K × 8 SRAMs

<sup>3</sup>SIMMs populated with 512K × 8 SRAMs

(continued on next page)

## 2.1 Configuration Jumpers

**Table 2–1 (Cont.) Configuration Jumper Position Descriptions**

Feature	Jack/Jumper—Pins and Description																																				
BC_SPEED<2:0>	<p>J1—17/18 (CONF7), —19/20 (CONF8), —21/22 (CONF9)</p> <p>These jumpers select the Bcache timing parameters used to compute the BC_CONFIG register value. Select the jumper configuration that matches the access time for the SRAMs being used.</p> <table border="1"> <thead> <tr> <th>CONF7 Pins 17/18</th> <th>CONF8 Pins 19/20</th> <th>CONF9 Pins 21/22</th> <th>Bcache Speed</th> </tr> </thead> <tbody> <tr> <td>In</td> <td>In</td> <td>In</td> <td>Reserved</td> </tr> <tr> <td>In</td> <td>In</td> <td>Out</td> <td>6-ns SRAM timing</td> </tr> <tr> <td>In</td> <td>Out</td> <td>In</td> <td>8-ns SRAM timing</td> </tr> <tr> <td>In</td> <td>Out</td> <td>Out</td> <td>10-ns SRAM timing (default)</td> </tr> <tr> <td>Out</td> <td>In</td> <td>In</td> <td>12-ns SRAM timing</td> </tr> <tr> <td>Out</td> <td>In</td> <td>Out</td> <td>15-ns SRAM timing</td> </tr> <tr> <td>Out</td> <td>Out</td> <td>In</td> <td>Reserved</td> </tr> <tr> <td>Out</td> <td>Out</td> <td>Out</td> <td>Reserved</td> </tr> </tbody> </table>	CONF7 Pins 17/18	CONF8 Pins 19/20	CONF9 Pins 21/22	Bcache Speed	In	In	In	Reserved	In	In	Out	6-ns SRAM timing	In	Out	In	8-ns SRAM timing	In	Out	Out	10-ns SRAM timing (default)	Out	In	In	12-ns SRAM timing	Out	In	Out	15-ns SRAM timing	Out	Out	In	Reserved	Out	Out	Out	Reserved
CONF7 Pins 17/18	CONF8 Pins 19/20	CONF9 Pins 21/22	Bcache Speed																																		
In	In	In	Reserved																																		
In	In	Out	6-ns SRAM timing																																		
In	Out	In	8-ns SRAM timing																																		
In	Out	Out	10-ns SRAM timing (default)																																		
Out	In	In	12-ns SRAM timing																																		
Out	In	Out	15-ns SRAM timing																																		
Out	Out	In	Reserved																																		
Out	Out	Out	Reserved																																		
Mini-Debugger	<p>J1—23/24 (CONF10)</p> <p>The Alpha SROM Mini-Debugger is provided in the SROM. This jumper (In) causes the SROM initialization to trap to the Mini-Debugger (connector J13) after all initialization is complete, but before starting the execution of the system ROM code. The default position for this jumper is out.</p>																																				

(continued on next page)

## 2.1 Configuration Jumpers

**Table 2–1 (Cont.) Configuration Jumper Position Descriptions**

Feature	Jack/Jumper—Pins and Description						
BOOT_OPTION	<p>J1—25/26 (CONF11)</p> <p>This jumper selects the image to be loaded into memory from the system flash ROM. With the jumper out (bit = 1), the first image (debug monitor) is loaded. With the jumper in (bit = 0), alternate images can be loaded depending upon the value stored in TOY RAM location 0x3F. The default position for this jumper is out.</p> <p>For more information on the SROM header and boot images, refer to Section B.8.</p>						
BC_RD_FAST	<p>J1—29/30 (CONF13)</p> <p>This jumper forces a Bcache read speed setting of 1 cycle faster than nominal.</p> <table border="1"> <thead> <tr> <th>BC_RD_FAST</th> <th>Bcache Speed</th> </tr> </thead> <tbody> <tr> <td>In</td> <td>Make read speed 1 cycle faster</td> </tr> <tr> <td>Out</td> <td>Nominal read speed (default)</td> </tr> </tbody> </table>	BC_RD_FAST	Bcache Speed	In	Make read speed 1 cycle faster	Out	Nominal read speed (default)
BC_RD_FAST	Bcache Speed						
In	Make read speed 1 cycle faster						
Out	Nominal read speed (default)						
Flash ROM write-protect/write-enable jumper	<p>J14</p> <table border="1"> <thead> <tr> <th>Jumper Pins</th> <th>Function</th> </tr> </thead> <tbody> <tr> <td>1 to 2</td> <td>Flash ROM write-protect</td> </tr> <tr> <td>2 to 3</td> <td>Flash ROM write-enable (default)</td> </tr> </tbody> </table>	Jumper Pins	Function	1 to 2	Flash ROM write-protect	2 to 3	Flash ROM write-enable (default)
Jumper Pins	Function						
1 to 2	Flash ROM write-protect						
2 to 3	Flash ROM write-enable (default)						

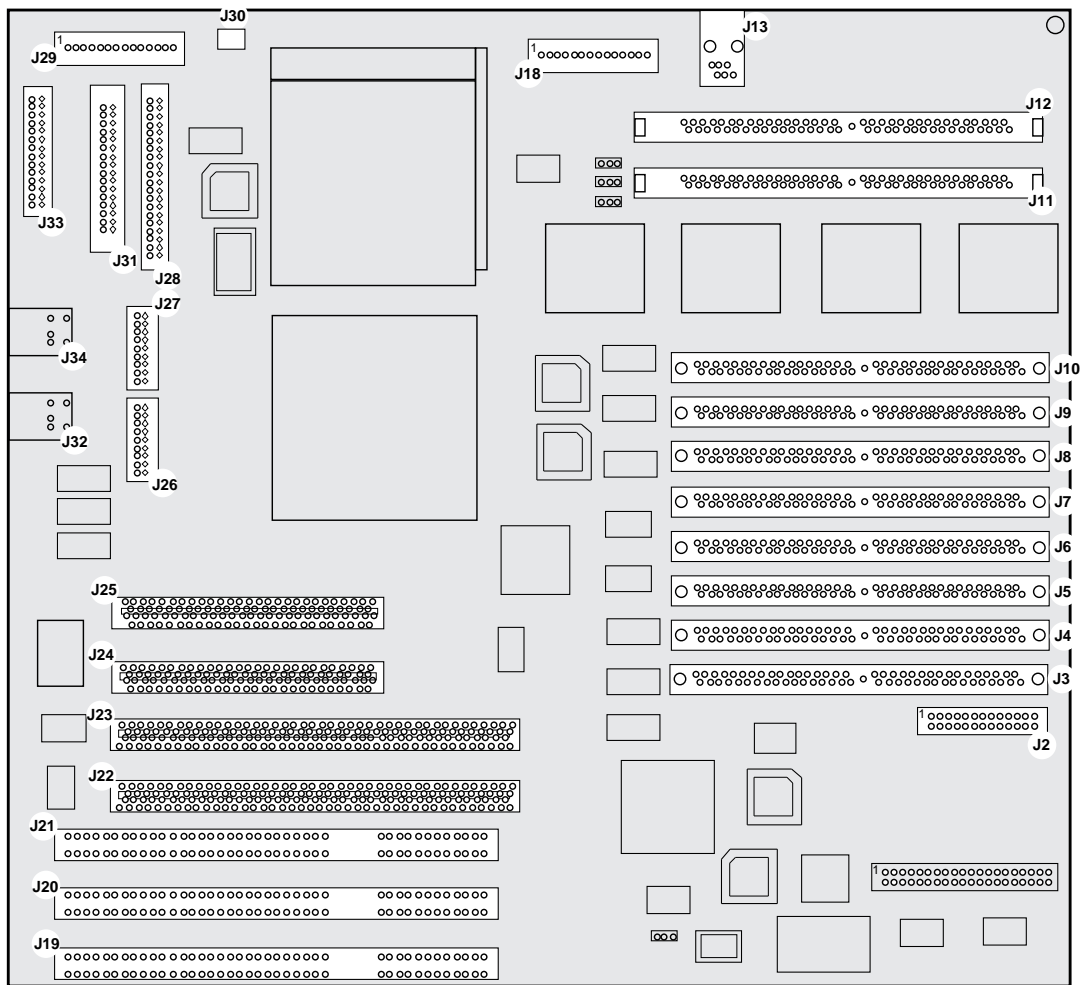
## **2.2 Alpha PCI 164-266/300 Connectors**

### **2.2 Alpha PCI 164-266/300 Connectors**

Figure 2-3 shows the Alpha PCI 164-266/300 connectors and Table 2-2 describes them. Figure 2-4 provides a detail of header connector J2.

## 2.2 Alpha PCI 164-266/300 Connectors

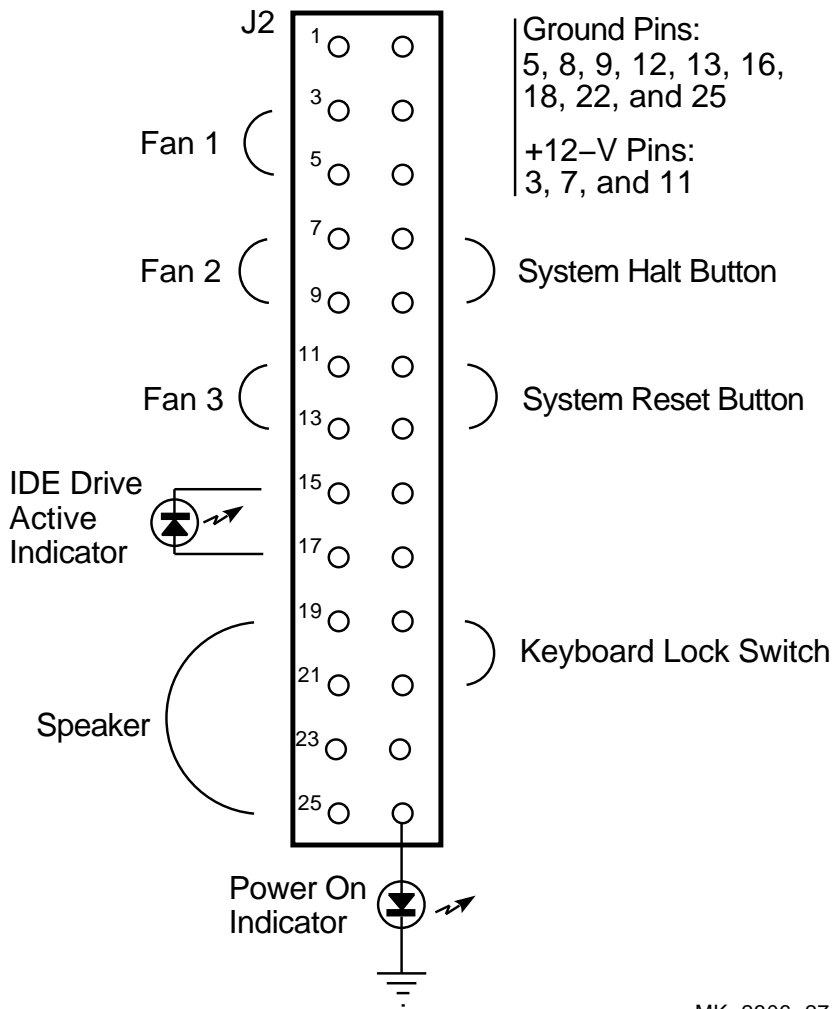
Figure 2-3 Alpha PCI 164-266/300 Connector Locations



MK-2306-31

## 2.2 Alpha PCI 164-266/300 Connectors

Figure 2-4 Detail of Header Connector J2



MK-2306-27

## 2.2 Alpha PCI 164-266/300 Connectors

Table 2–2 Alpha PCI 164-266/300 Connector Descriptions

Connector	Pins	Description
<b>Main Memory/Bcache SIMMS</b>		
J10	72	DRAM 0 SIMM
J9	72	DRAM 1 SIMM
J8	72	DRAM 2 SIMM
J7	72	DRAM 3 SIMM
J6	72	DRAM 4 SIMM
J5	72	DRAM 5 SIMM
J4	72	DRAM 6 SIMM
J3	72	DRAM 7 SIMM
<b>Note:</b> To fill a 256-bit data path, all SIMM connectors J3 through J10 must be populated.		
J11	60	Bcache 0 SIMM
J12	60	Bcache 1 SIMM
<b>Note:</b> Both Bcache SIMM connectors must be populated.		
<b>PCI Slots</b>		
J22	184	PCI64 slot 0
J23	184	PCI64 slot 1
J24	124	PCI32 slot 2
J25	124	PCI32 slot 3
<b>ISA Slots</b>		
J19	98	ISA slot 0
J20	98	ISA slot 1
J21	98	ISA slot 2
<b>Keyboard Connector</b>		
J32	6	Keyboard connector

(continued on next page)

## 2.2 Alpha PCI 164-266/300 Connectors

**Table 2–2 (Cont.) Alpha PCI 164-266/300 Connector Descriptions**

Connector	Pins	Description
		<b>Mouse Connector</b>
J34	6	Mouse connector
		<b>National 87312 Combination Chip Connectors</b>
J33	26	Parallel port connector
		Connects to an external 25-pin connector.
J27	10	Serial communication port 1 connector
		<b>Note:</b> This connector can be used as a terminal port for the debug monitor.
J26	10	Serial communication port 2 connector
J31	34	Diskette drive connector
J28	40	IDE drive connector
		<b>SROM Data/Clock</b>
J13	6	SROM data/clock serial port input connector
		<b>Note:</b> This connector can be used as a terminal port for the Mini-Debugger.
J2	26	Header connector J2 is a straight double-row header with standard 0.025-in pins on 0.10-in centers. Connections to it may be made by means of individual 2- or 4-pin female plugs. Figure 2–4 provides a detail of header connector J2.
		<b>System Enclosure Fans</b>
J2–3/5, –7/9, –11/13	2 each	Up to three 12-V cooling fans may be connected to these pins.
		<b>IDE Drive Active Indicator</b>
J2–15/17	2	IDE drive active indicator pins

(continued on next page)



## 2.2 Alpha PCI 164-266/300 Connectors

Table 2-2 (Cont.) Alpha PCI 164-266/300 Connector Descriptions

Connector	Pins	Description
		<b>Speaker</b>
J2-19/21/23/25	—	Speaker connector pins
		<b>Power On Indicator</b>
J2-26	1	Power on indicator pin Connect LED from this pin to ground.
		<b>System Halt Button</b>
J2-8/10	2	System halt button pins
		<b>System Reset Button</b>
J2-12/14	2	System reset button pins
		<b>Keyboard Lock Switch</b>
J2-20/22	2	Keyboard lock switch pins

(continued on next page)

## 2.2 Alpha PCI 164-266/300 Connectors

Table 2–2 (Cont.) Alpha PCI 164-266/300 Connector Descriptions

Connector	Pins	Description	
J18	12	<b>Power Connectors</b>	
		Board power connector <sup>1</sup>	
		<b>Pin</b>	<b>Voltage/Signal</b>
		1	+3.3 V
		2	+3.3 V
		3	+3.3 V
		4	Ground
		5	Ground
		6	Ground
		7	Ground
		8	Ground
		9	Ground
		10	+3.3 V
11	+3.3 V		
12	+3.3 V		

<sup>1</sup>Typically mates with two 6-pin power supply connectors.

(continued on next page)

## 2.2 Alpha PCI 164-266/300 Connectors

Table 2–2 (Cont.) Alpha PCI 164-266/300 Connector Descriptions

Connector	Pins	Description	
J29	12	Board power connector <sup>1</sup>	
		<b>Pin</b>	<b>Voltage/Signal</b>
		1	<b>p_dcok</b>
		2	+5 V
		3	+12 V
		4	–12 V
		5	Ground
		6	Ground
		7	Ground
		8	Ground
		9	–5 V
		10	+5 V
		11	+5 V
12	+5 V		
J30	3	CPU fan power and sensor	
		<p><b>Caution: Fan sensor required.</b></p> <p>The fan <i>must</i> have a built-in sensor that drives a signal low while the fan spins. The sensor must be connected to pin J30–2. The fan supplied with the Alpha PCI 164-266/300 includes an airflow sensor.</p>	

<sup>1</sup>Typically mates with two 6-pin power supply connectors.

## 2.2 Alpha PCI 164-266/300 Connectors

**Table 2-3 List of Connector and Jumper Part Numbers**

Designation	Description	Vendor Part Number or Equivalent
<b>Note:</b> Specific vendor part numbers are for reference only and should not be construed as a recommendation by Digital.		
J1	(Jumpers)	AMP 531220-3
J14	Flash ROM write enable (jumpers)	AMP 531220-3
J15	(jumpers)	AMP 531220-3
J16	(jumpers)	AMP 531220-3
J17	(jumpers)	AMP 531220-3
J18	Power (connector)	Molex 87218-1205
J19-J21	ISA slots (connector)	AMP 645169-3
J26	Serial port 2 (connector)	DUPONT 65611-110
J27	Serial port 1 (connector)	DUPONT 65611-110
J28	IDE drive (connector)	DUPONT 66506-354
J29	Power (connector)	Molex 87218-1202
J31	Diskette drive (connector)	DUPONT 66506-353
J32	Keyboard (connector)	HON HAI PRES MH11067-H1
J33	Parallel port (connector)	AMP 1-10386-3
J34	Mouse (connector)	HON HAI PRES MH11067-H1

# 3

---

## Functional Description

This chapter describes the functional operation of the Alpha PCI 164-266/300. The description introduces the DECchip 21171 ASIC support chipset and describes its implementation with the 21164 microprocessor, its supporting memory, and I/O devices. Figure 1-1 shows the Alpha PCI 164-266/300 major functional components.

Information, such as bus timing and protocol, found in other data sheets and reference documentation is not duplicated. See Appendix C for a list of supporting documents and order numbers.

---

### Note

---

For detailed descriptions of bus transactions, chipset logic, and operation, refer to the *Alpha 21164 Microprocessor Hardware Reference Manual* and the *DECchip 21171 Core Logic Chipset Technical Reference Manual*.

For details of the PCI interface, refer to the *PCI System Design Guide*.

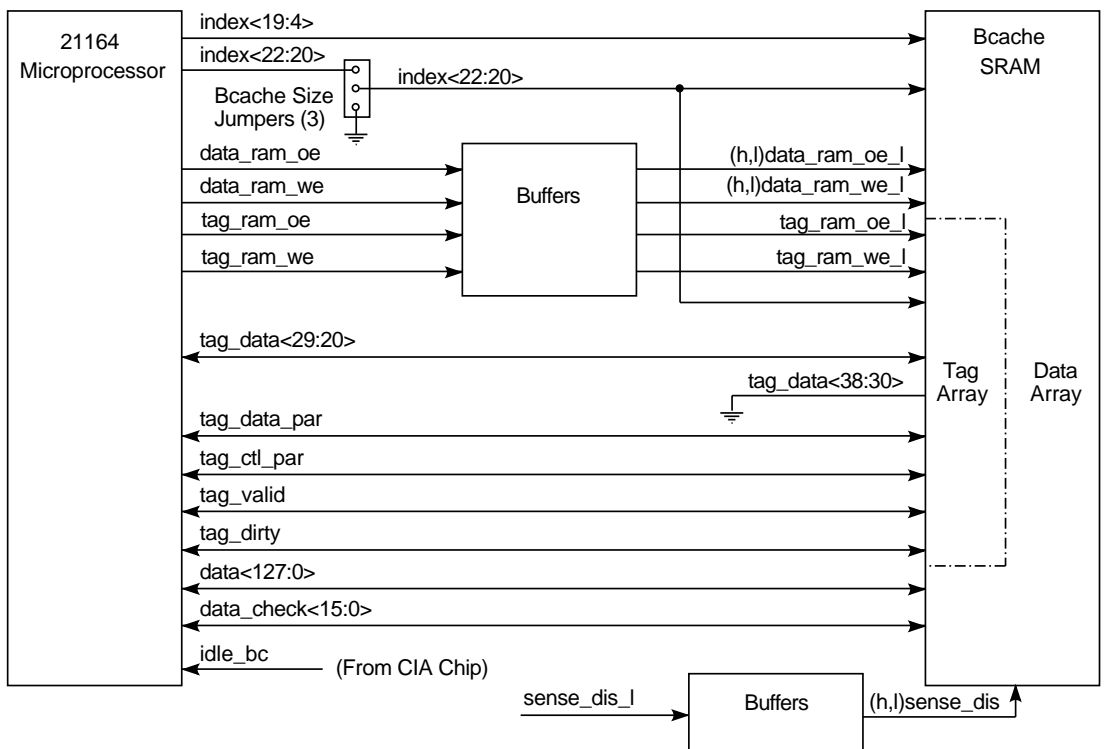
---

### 3.1 Alpha PCI 164-266/300 Bcache Interface

### 3.1 Alpha PCI 164-266/300 Bcache Interface

The 21164 controls the backup cache (Bcache) array (see Figure 3-1). The data bus (**data<127:0>**), check bus (**data\_check<15:0>**), **tag\_dirty**, and **tag\_ctl\_par** signals are shared with the system interface.

Figure 3-1 Alpha PCI 164-266/300 Bcache Array



SW-2306-18

### 3.1 Alpha PCI 164-266/300 Bcache Interface

The Bcache interface supports multiple cache sizes and access times. The cache sizes supported are:

- 2MB with Alpha cache inline memory modules (DIMMs) populated with 128K × 8 static RAMs (SRAMs)
- 2MB, 4MB, and 8MB with DIMMs populated with 512K × 8 SRAMs

SRAM speeds of 6 ns to 15 ns can be used. In most cases, wave pipelining can decrease the cache loop times by one CPU cycle. Performance trade-offs for each application can then be made between size, speed, and Bcache expense.

Because of the support for smaller Bcache sizes, larger cache sizes contain extra tag bits. Alpha PCI 164-266/300 modules fitted with large caches can be configured to operate in any of the smaller cache sizes because the extra tag bits are available. Caches fitted with 512K × 8 SRAM DIMMs support all the cache sizes. Caches fitted with 128K × 8 SRAM DIMMs only support the 2MB configuration. Table 3–1 lists the three supported Bcache configurations.

**Table 3–1 Bcache Configurations**

Cache Size	Block Size	Index	Tag	Control	Data	ECC
8MB	64-byte	<22:4>	<29:23>,P	V,D,P	<127:0>	<15:0>
4MB	64-byte	<21:4>	<29:22>,P	V,D,P	<127:0>	<15:0>
2MB	64-byte	<20:4>	<29:21>,P	V,D,P	<127:0>	<15:0>

**Key to control bits**

V = Valid  
D = Dirty  
P = Parity

The Alpha PCI 164-266/300 Bcache operates only in 64-byte mode because the data switch (DSW) chip only supports 64-byte transfers to and from memory. Wave pipelined accesses to the Bcache are supported.

Buffers are required between the 21164 and the Bcache SRAMs on the output-enable and write-enable signals for the data and tags. These buffers provide the required inversion for the enable signals as well as the load buffering to drive the multiple SRAMs.

## 3.2 DECchip 21171 Chipset

### 3.2 DECchip 21171 Chipset

The DECchip 21171 chipset provides a cost-competitive solution for designers using the 21164 microprocessor to develop uniprocessor systems. The chipset provides a 256-bit memory interface and a peripheral component interconnect (PCI) I/O interface, and includes the following two gate arrays:

- DECchip 21171-CA control, I/O interface, and address (CIA) chip packaged in a 383-pin, pin grid array (PGA)
- DECchip 21171-BA data switch (DSW) chip packaged in a 208-pin plastic quad flat pack (PQFP)

Figure 3–2 shows the Alpha PCI 164-266/300 implementation of the DECchip 21171 chipset.

#### 3.2.1 CIA Chip Overview

The control, I/O interface, and addressing (CIA) application-specific integrated circuit (ASIC) accepts addresses and commands from the 21164 and drives the main memory array with the address, and control signals. It also provides an interface to the 64-bit PCI I/O bus.

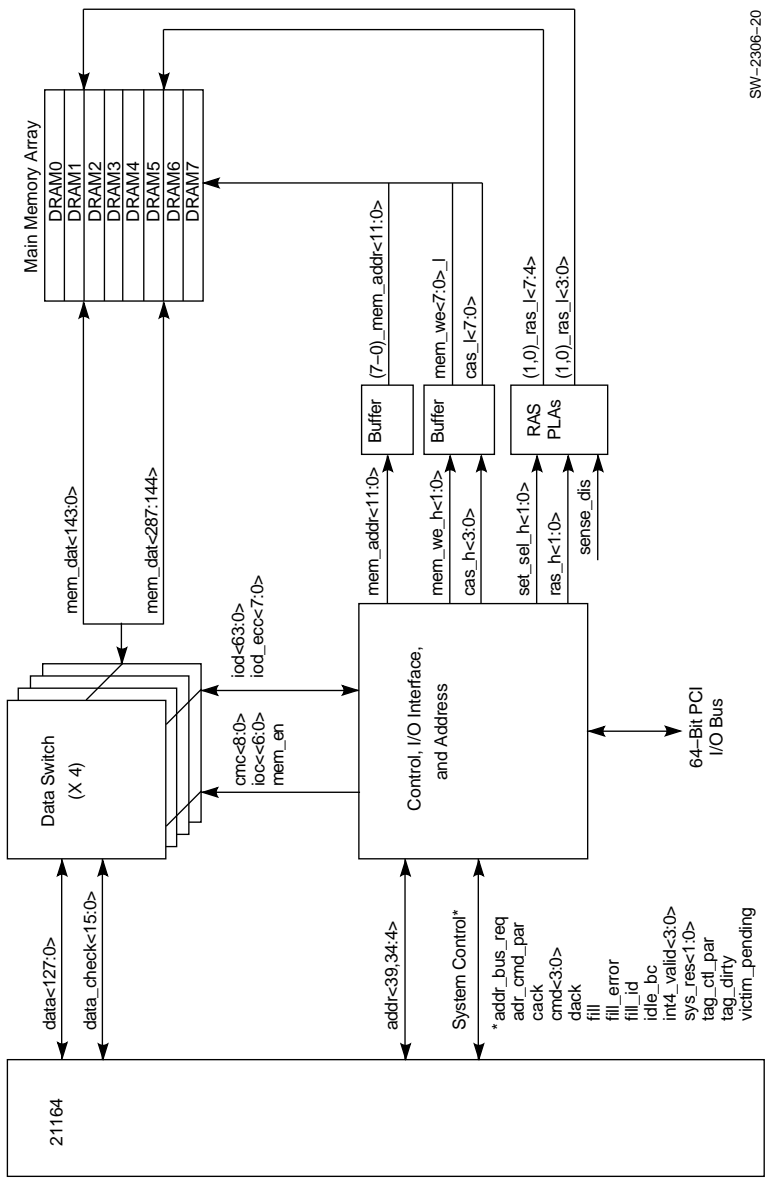
The CIA chip provides the following functions:

- Serves as the interface between the 21164, main memory (addressing and control), and the PCI bus. A 3-entry CPU instruction queue is implemented to capture commands should the memory or I/O port be busy.
- Provides the DSW chips with control information to direct the data flow.
- Provides the interface to the PCI bus, and contains a portion of the data path. This includes the error correction code (ECC) generation and check logic for data transfers to and from the DSW chips. It also contains data buffers for all four transaction types (I/O read and write operations, and direct memory access (DMA) read and write operations). Each buffer is 64 bytes in size.
- Generates the row and column addresses for the dynamic random-access memory (DRAM) SIMMs, as well as all the memory control signals (RAS, CAS, WE). All the required DRAM refresh control is contained in the CIA.
- Provides all the logic to map 21164 noncacheable addresses to PCI address space, as well as all the translation logic to map PCI DMA addresses to system memory.



### 3.2 DECchip 21171 Chipset

Figure 3-2 Main Memory Interface



SW-2306-20

## 3.2 DECchip 21171 Chipset

Two DMA conversion methods are supported: direct mapping, where a base offset is concatenated with the PCI address, and scatter-gather mapping, which maps an 8KB PCI page to any 8KB memory page. The CIA contains an 8-entry scatter-gather translation lookaside buffer (TLB), where each entry holds four consecutive page table entries (PTEs).

Refer to Chapter 4 for additional details on PCI and DMA address mapping.

### 3.2.2 DSW Overview

Four data switch (DSW) ASICs provide the interface between the 128-bit 21164 data bus (**data<127:0>**) and 16-bit check bus (**data\_check<15:0>**), the 288-bit DRAM memory data bus (**mem\_dat<287:0>**), and the CIA chip for PCI data (**iod<63:0>** and **iod\_ecc<7:0>**). The DSW chips (four required) provide the system with a 256-bit-wide memory path.

The DSW chip contains the memory interface data path. This includes a 64-byte victim buffer, a 32-byte I/O read buffer, four 32-byte I/O write buffers, and two DMA buffers.

The four DSW chips receive data from the CPU by means of the 128-bit CPU data bus. They transfer data to and from the CIA by means of the 64-bit IOD bus. Any data directed to or from the PCI bus must be transferred through the CIA. The DSW chips also provide an interface to the 256-bit-wide memory data bus.

## 3.3 Main Memory Interface

Four DSW chips, along with the CIA, provide a 256-bit-wide, high-speed memory data path for both CPU memory accesses and PCI DMA. The Alpha PCI 164-266/300 supports and requires eight DRAM SIMM modules. The DSWs are configured to run in 256-bit mode.

Quadword ECC is supported on the DRAM and CPU buses. The same quadword ECC that is supported by the 21164 is also supported on the memory bus. Byte parity is generated on the PCI bus. Only 64-byte transfers are supported because the DSW chips do not support the 21164's optional 32-byte mode.

The Alpha PCI 164-266/300 supports a maximum of 512MB of main memory. In all cases, the memory is organized as one single bank. All CPU cacheable memory accesses and PCI DMA accesses are controlled and routed to main memory by the DECchip 21171 chipset.

The Alpha PCI 164-266/300 supports the following SIMM sizes:

- 1MB × 36-bit DRAM SIMM

### 3.3 Main Memory Interface

- 2MB × 36-bit DRAM SIMM
- 4MB × 36-bit DRAM SIMM
- 8MB × 36-bit DRAM SIMM
- 16MB × 36-bit DRAM SIMM

The following memory sizes are supported with one set of eight SIMMs:

- 32MB memory
- 64MB memory
- 128MB memory
- 256MB memory
- 512MB memory

The row and column addresses for the DRAM SIMMs are partitioned such that any victim's row address will match its corresponding read miss's row address. This allows a page-mode-write operation to follow a read operation during read miss/victim processing.

### 3.4 PCI Devices

The Alpha PCI 164-266/300 uses the PCI bus as the main I/O bus for the majority of peripheral functions. The board implements the ISA bus as an expansion bus for system support functions and relatively slow peripheral devices.

The PCI bus supports multiplexed, burst mode, read and write transfers. It supports synchronous operation of between 25 MHz and 33 MHz. It also supports either a 32-bit or 64-bit data path with 32-bit device support in the 64-bit configuration. Depending upon the configuration and operating frequencies, the PCI bus supports anywhere between 100MB/s (25 MHz, 32-bit) to 264MB/s (33 MHz, 64-bit) peak throughput. The PCI provides parity on address and data cycles. Three physical address spaces are supported:

1. 32-bit memory space
2. 32-bit I/O space
3. 256-byte-per-agent configuration space

## 3.4 PCI Devices

The bridge from the 21164 system bus to the 64-bit PCI bus is provided by the CIA chip. It generates the required 32-bit PCI address for 21164 I/O accesses directed to the PCI. It also accepts 64-bit double address cycles and 32-bit single address cycles. However, the 64-bit address support is subject to some constraints. Refer to Chapter 4 for more information on these constraints.

### 3.4.1 Saturn-IO (SIO) Chip

To provide the Alpha PCI 164-266/300 with greater flexibility, the only embedded PCI device is the SIO PCI-to-ISA chip. All other functions are provided by option modules. The 82378ZB SIO chip provides the bridge between the PCI bus and the Industry Standard Architecture (ISA) bus. The SIO incorporates the logic for:

- A PCI interface (master and slave)
- An ISA interface (master and slave)
- Enhanced 7-channel DMA controller that supports fast DMA transfers and scatter-gather, and data buffers to isolate the PCI bus from the ISA bus
- PCI and ISA arbitration
- A 14-level interrupt controller
- A 16-bit basic input/output system (BIOS) timer
- Three programmable timer counters
- Nonmaskable interrupt (NMI) control logic
- Decoding and control for utility bus peripheral devices
- Speaker driver

Refer to Intel document *82420/82430 PCIset ISA and EISA Bridges* for additional information.

### 3.4.2 PCI Expansion Slots

Three dedicated PCI expansion slots are provided on the Alpha PCI 164-266/300, as well as one combination slot that can be used for either ISA or PCI expansion. They use the standard 5-V PCI connector and pinout for both the 32-bit and the 64-bit implementations. This allows the system user to add additional 32-bit or 64-bit PCI options. The 3.3 V is not supplied to any PCI connectors. The SIO chip provides the interface to the ISA expansion I/O bus.

## 3.5 ISA Bus Devices

### 3.5 ISA Bus Devices

Figure 3–3 shows the Alpha PCI 164-266/300 ISA bus implementation with peripheral devices and connectors. Two dedicated ISA expansion slots are provided in addition to the combination ISA/PCI slot. System support features such as serial lines, parallel port, integrated drive electronics (IDE), and diskette controller are embedded on the module by means of an 87312 combination controller chip. Also shown is the utility bus (Ubus) with its system support devices.

#### 3.5.1 Combination Controller

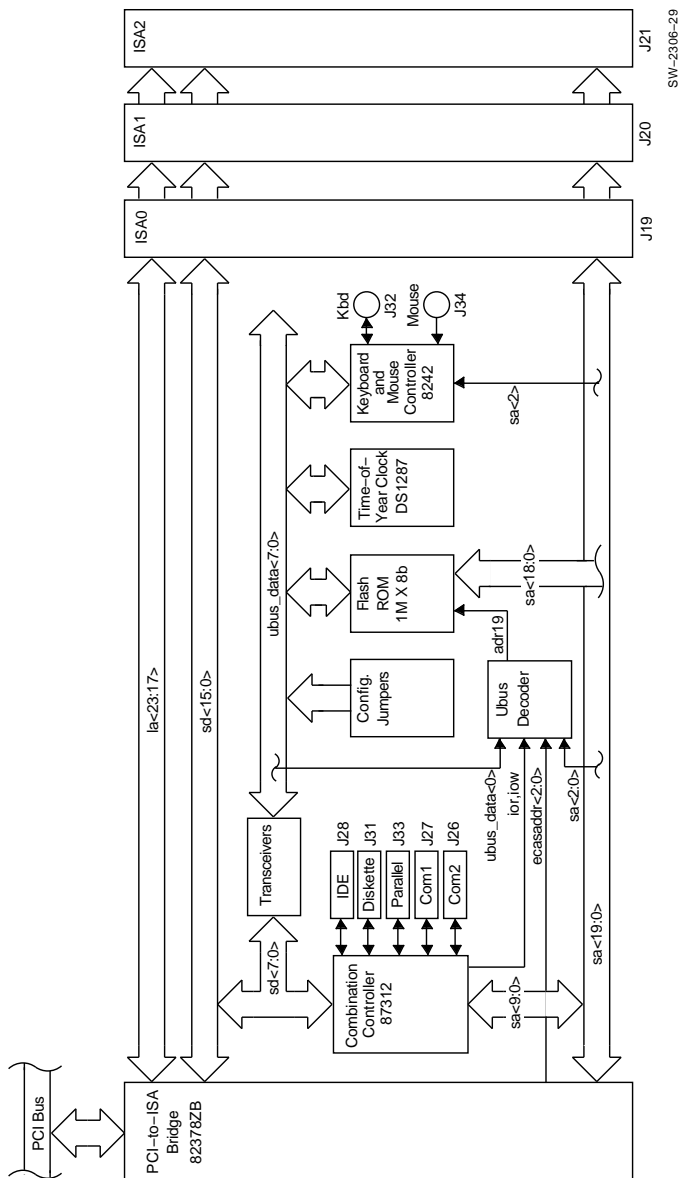
The Alpha PCI 164-266/300 uses the National Semiconductor PC87312 as the combination controller chip. (See Figure 3–3.) It is packaged in a 100-pin PQFP configuration. The chip provides the following ISA peripheral functions:

- Integrated drive electronics (IDE) controller—Provides a complete IDE interface. Signal buffers are provided on the Alpha PCI 164-266/300. The interface is brought out to a standard 40-pin header. A ribbon cable connects the header to one or two IDE drives. An “IDE Drive Active” indicator can be connected to header J2 (see Figure 2–4).
- Diskette controller—Software compatible to the Intel PC8477 (contains a superset of the Intel DP8473 and NEC  $\mu$ PD765) and the Intel N82077 FDC functions. The onchip analog data separator requires no external filter components and supports the 4MB drive format and other standard diskette drives used with 5.25-inch and 3.5-inch media. FDC data and control lines are brought out to a standard 34-pin connector. A ribbon cable interfaces the connector to one or two diskette drives.
- Serial ports—Two universal asynchronous receiver–transmitters (UARTs) with full modem control, compatible with NS16450 or PC16550 devices, are brought out to separate onboard, 10-pin connectors. The lines can be brought out through 25-pin female D-sub connectors on the bulkhead of a standard PC enclosure.
- Parallel port—The bidirectional parallel port is brought out to an onboard 26-pin connector, J33. It can be brought out through a 25-pin female D-sub connector on the bulkhead of a standard PC enclosure.

An onboard 9154-10 clock generator chip supplies a 24-MHz reference clock for the diskette data separator and serial ports.

### 3.5 ISA Bus Devices

Figure 3-3 ISA Devices



### 3.5 ISA Bus Devices

Refer to National Semiconductor document *PC87311/PC87312 (SuperI/O™ II/III) Floppy Disk Controller with Dual UARTs, Parallel Port, and IDE Interface* for additional information.

## 3.5 ISA Bus Devices

### 3.5.2 Keyboard and Mouse Controller

The Intel N8242 located on the ISA utility bus provides the keyboard and mouse controller functions. It is packaged in a 44-pin plastic leadless chip carrier (PLCC) configuration.

The 8242 is an Intel UPI™-42AH universal peripheral interface. It is an 8-bit slave microcontroller with 2KB of ROM and 256 bytes of RAM that has been preprogrammed with a keyboard BIOS for standard scan codes.

Refer to Intel document *UPI™-41AH/42AH Universal Peripheral Interface 8-Bit Slave Microcontroller* for additional information.

### 3.5.3 Time-of-Year Clock

The Dallas Semiconductor DS1287 chip, located on the ISA utility bus, provides the time-of-year (TOY) function. It is packaged in a plastic 24-pin DIP configuration. The DS1287 is designed with onchip RAM, a lithium battery, a quartz crystal, and write-protection circuitry. (See Figure 3–3.)

The functions available to the user include:

- A nonvolatile time-of-day clock
- An alarm
- A 100-year calendar
- Programmable interrupt ability
- A square-wave generator
- 50 bytes of nonvolatile SRAM

Contents of the time-of-day register and memory are maintained in the absence of power through the lithium battery.

The DS1287 includes three separate, fully automatic interrupt sources for a processor. The alarm interrupt can be programmed to occur at rates from one per second to one per day. The periodic interrupt can be selected for rates from 122  $\mu$ s to 500 ms. The update-ended interrupt can be used to indicate to the program that an update cycle has completed. The device interrupt line is presented to the system interrupt multiplexer.



## 3.5 ISA Bus Devices

### 3.5.4 Utility Bus Memory Device

The Alpha PCI 164-266/300 utility bus (Ubus) drives a flash ROM memory device. The flash ROM chip provides 1MB of flash memory for operating system support.

Flash data is accessed through 20 address inputs. The low-order 19 address bits are driven by ISA bus **sa<18:0>**. The high-order 20th bit (**flash\_adr19**) is driven by the Ubus decode PAL. Address bit **flash\_adr19** can be changed by writing to ISA I/O port x800.

The +12 V is applied to the flash ROM by means of Jumper J14 so that code updates can be accomplished, if desired.

### 3.5.5 ISA Expansion Slots

Three ISA expansion slots are provided for plug-in ISA peripherals. One of the slots is shared with the 64-PCI and can be used for a PCI or ISA device.

## 3.6 Interrupts

### 3.6 Interrupts

This section describes the Alpha PCI 164-266/300 interrupt logic. PCI-, ISA-, and CIA-generated interrupts are each described. Figure 3-4 shows the interrupt logic.

The PCI-to-ISA SIO bridge chip provides the functionality of two 8259 interrupt control devices. These ISA-compatible interrupt controllers are cascaded such that 14 external and two internal interrupts are available. The PCI interrupt acknowledge command should be used to read the interrupt request vector from the SIO.

However, the Alpha PCI 164-266/300 has more interrupt signals than the 14 external interrupts the SIO can handle. Therefore, all the ISA interrupts are sent to the SIO except for the two CIA interrupts, the time-of-year (TOY) interrupt, and the 16 PCI interrupts. They are sent to an external interrupt PAL. This PAL takes these interrupts, as well as an OR of the nonexistent memory (NMI) and error signals from the SIO, and generates **cpu\_irq<3:0>**. During reset, **cpu\_irq<3:0>** convey the system clocking ratios and delays, which are set by jumpers on J1.

Table 3-2 lists each system interrupt, its fixed interrupt priority level (IPL), and its Alpha PCI 164-266/300 implementation. Table 3-3 lists each SIO interrupt and its Alpha PCI 164-266/300 implementation.

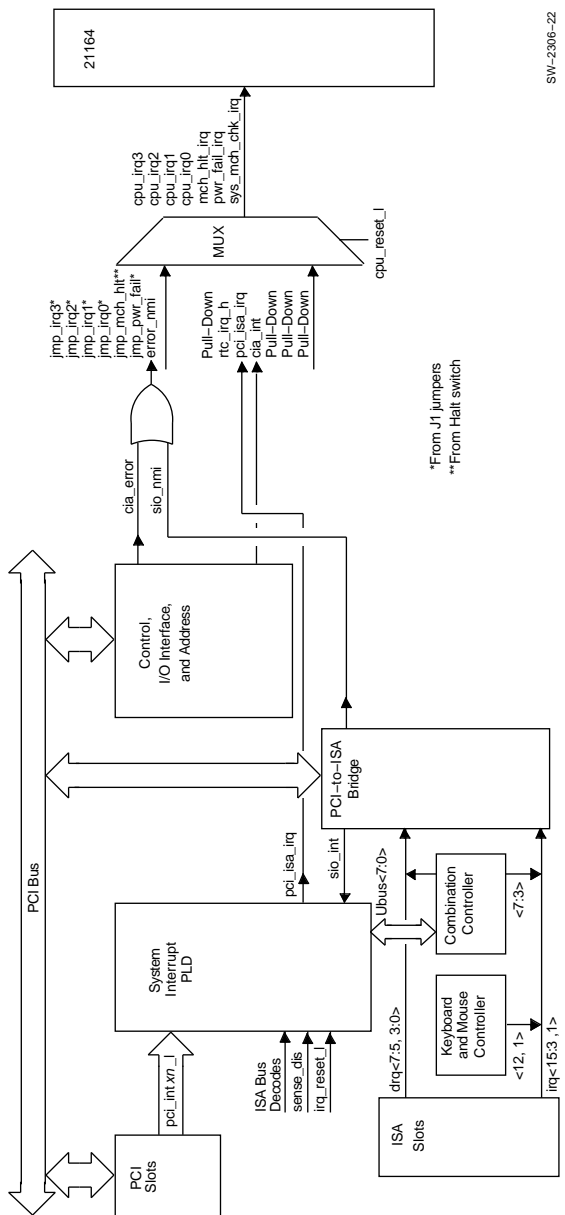
**Table 3-2 Alpha PCI 164-266/300 System Interrupts**

21164 Interrupt	IPL <sup>1</sup>	Suggested Usage	Alpha PCI 164-266/300 Usage
<b>cpu_irq&lt;0&gt;</b>	20	Corrected system error	Corrected ECC error and sparse space reserved encodings detected by CIA
<b>cpu_irq&lt;1&gt;</b>	21	—	PCI and ISA interrupts
<b>cpu_irq&lt;2&gt;</b>	22	Interprocessor and timer interrupts	Time-of-year clock interrupt
<b>cpu_irq&lt;3&gt;</b>	23	—	Reserved
<b>pwr_fail_irq</b>	30	Powerfail interrupt	Reserved
<b>sys_mch_chk_irq</b>	31	System machine check interrupt	SIO NMI and CIA errors
<b>mch_hlt_irq</b>	—	Halt	Reserved

<sup>1</sup>IPL = interrupt priority level (fixed)

### 3.6 Interrupts

Figure 3-4 Interrupt Logic



SW-2306-22

## 3.6 Interrupts

**Table 3–3 PCI-to-ISA SIO Bridge Interrupts**

Priority	Label	Controller	Internal/External	Interrupt Source
1	IRQ0	1	Internal	Internal timer 1
2	IRQ1	1	External	Keyboard
3–10	IRQ2	1	Internal	Interrupt from controller 2
3	IRQ8# <sup>1</sup>	2	External	Reserved
4	IRQ9	2	External	ISA bus pin B04
5	IRQ10	2	External	ISA bus pin D03
6	IRQ11	2	External	ISA bus pin D04
7	IRQ12	2	External	Mouse
8	IRQ13	2	External	Reserved
9	IRQ14	2	External	IDE
10	IRQ15	2	External	ISA bus pin D06
11	IRQ3	1	External	87312 combination controller
12	IRQ4	1	External	87312 combination controller
13	IRQ5	1	External	87312 combination controller
14	IRQ6	1	External	87312 combination controller
15	IRQ7	1	External	87312 combination controller

<sup>1</sup>The # symbol indicates an active low signal.

### 3.7 System Clocks

Figure 3–5 shows the Alpha PCI 164-266/300 clock generation and distribution scheme.

The Alpha PCI 164-266/300 board includes input clocks to the microprocessor as well as clock distribution for the various system memory and I/O devices. There are other miscellaneous clocks for ISA bus support. System clocking can be divided into three main areas:

- Microprocessor input clock

The input clock runs at twice the operating frequency of the 21164. The Alpha PCI 164-266/300 supports cycle times from 3.2 ns to 5.0 ns. This implies input clock frequencies from 400 MHz to 625 MHz. The clock is provided using a relatively low frequency oscillator whose output is fed into a phase-locked loop (PLL). The PLL multiplies the input frequency by a factor of 20. This 20× frequency is then used as the 21164 input clock.

- Clock distribution

Clock distribution includes the distribution of system clocks from the 21164 to the system logic. The Alpha PCI 164-266/300 clock distribution scheme is flexible enough to allow the majority of cycle times combinations to be supported. Because the PCI is synchronous to the system clock generated by the 21164, the PCI cycle time is a multiple of the 21164 cycle time. This distribution scheme allows a range of supported PCI clock combinations between 25 MHz and 33 MHz.

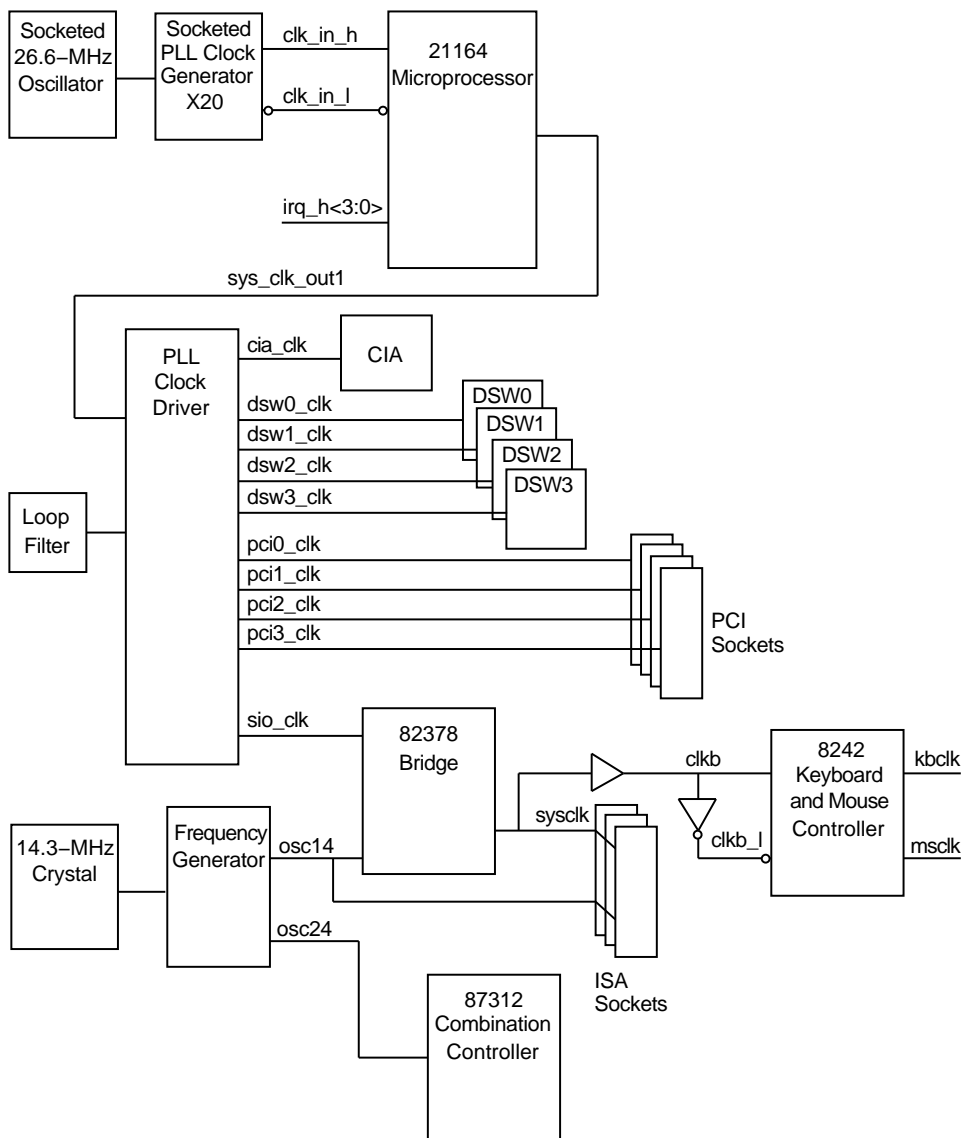
- Miscellaneous clocks

The miscellaneous clocks include those needed for ISA and the combination controller. These clocks are provided by a crystal and a frequency generator with fixed scaling.

The base microprocessor input clock oscillator runs at 26.66 or 30.00 MHz. A TriQuint TQ2061 phase-locked loop (PLL) multiplier synthesizes a higher frequency CPU clock (signals **clk\_in\_h,l**) and drives the 21164 differential clock inputs at 20 times the base clock rate. The 21164 microprocessor divides this clock by 2 to generate its internal 266.6 or 300.0 MHz clock. The divide-by-two function is set because the 21164's **clk\_mode\_h<1:0>** input pins are both grounded (normal mode). This oscillator is socketed.

### 3.7 System Clocks

Figure 3-5 System Clocks and Distribution



SW-2306-14

### 3.7 System Clocks

At system reset, the microprocessor's **irq\_h<3:0>** pins are driven by the clock divisor values set by four jumpers on J1. During normal operation, these signals are used for interrupt requests. The pins are either switched to ground or pulled up in a specific combination to set the 21164's internal divider. The divisor is programmable and can range from 3 to 15. (Refer to Table 2-1 for a list of jumper combinations.)

The 21164 microprocessor produces the divided clock output signal **sys\_clk\_out1** that drives the Motorola 88PL117 PLL clock driver chip. This synchronous system clock provides the system memory and I/O clock reference.

The clock driver chip is used to minimize system level clock skew as well as creating square-wave clocks from what is an asymmetrical clock from the 21164. The clock driver provides a 50% duty cycle output clock that is referenced to the 21164's **sys\_clk\_out1** and aligned with a reference feedback clock. The clock driver is configured (OPT<2:0> = 011) such that the output frequency equals the input frequency and is in phase. The PLL provides copies to each DSW chip, the CIA chip, each PCI slot, and the PCI-to-ISA bridge.

The DSW/CIA chipset generates its own 1X and 2X clocks on each ASIC. Each ASIC uses an integrated PLL together with an onchip clock trunk/buffer scheme to maintain chip skews under 0.6 ns.

Clock signal **sio\_clk** synchronizes the PCI-to-ISA bridge's PCI bus transactions. The supported PCI cycle times range from 40 ns (25 MHz) to 30 ns (33.3 MHz).

A 14.3-MHz crystal output is buffered through an AV9154-10 frequency generator to produce the signal **osc14**. Signal **osc14** is routed to the PCI-to-ISA bridge and the three ISA slots. This is the standard 14.31818-MHz ISA clock signal.

The frequency generator produces a 24.0-MHz clock (41.7-ns period) **osc24**. This signal provides clocking for the 87312 combination controller's IDE and diskette data separator.

## 3.8 Reset and Initialization

### 3.8 Reset and Initialization

A power monitor senses +3.3 V dc to ensure that it is stable before the 21164 CPU's inputs and I/O pins are driven (see Figure 3-6). Any device that drives the 21164 has a tristate output controlled by the power monitor output. This is necessary because the 21164 must not have its inputs driven to greater than 4.0 V if the 3.3-V level to the 21164 is not greater than 2.5 V. The TL7702B provides this function by sensing whether the 3.3-V level is above or below 2.5 V.

Should the +3.3 V dc supply fail, the power monitor enables **sense\_dis**, which is applied to the reset logic. The reset logic generates a group of reset functions to the 21164 and the remainder of the system.

An external reset switch can be connected to header J2. The reset function initializes the 21164 and the system logic, but does not send an initialization pulse to the ISA devices. The **p\_dcok** signal provides a full system initialization, equivalent to a power-down and power-up cycle.

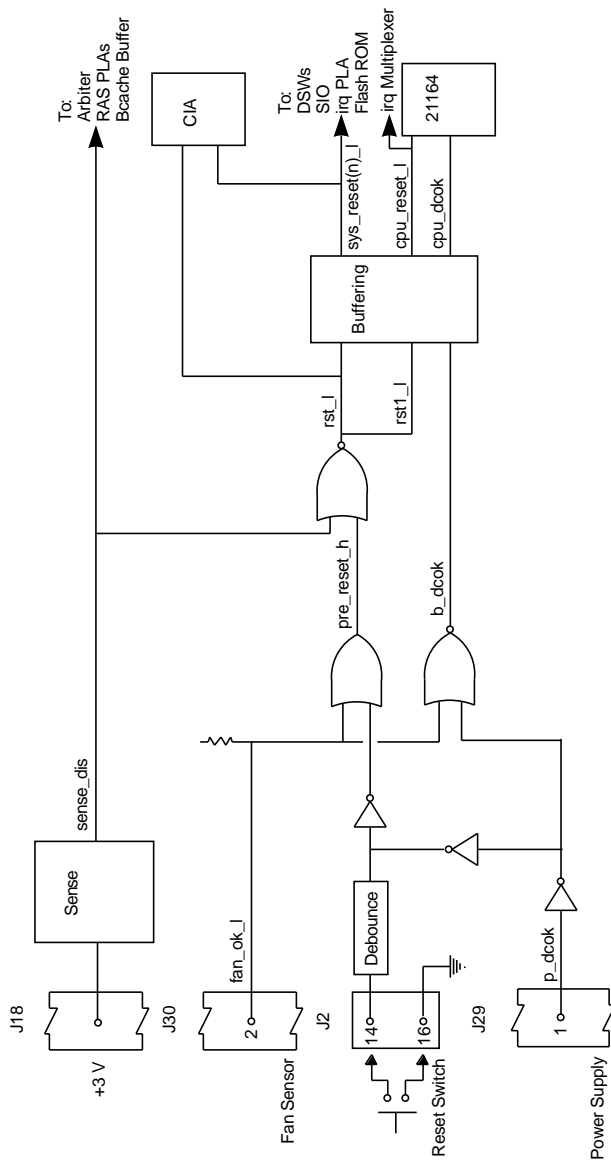
In addition, the fan sense signal (**fan\_ok\_1**) is logically ORed with the reset switch output and, when enabled, drives **cpu\_dcok** and **cpu\_reset\_1**, indicating a fan failure.

The **rst\_1** signal is buffered and drives a set of **sys\_reset** signals to reset the remainder of the system, including PCI and ISA devices through the CIA chip.



### 3.8 Reset and Initialization

Figure 3-6 System Reset and Initialization



SW-2306-16

## 3.9 Serial ROM

### 3.9 Serial ROM

The serial ROM (SROM) provides the following functions:

- Initializes the CPU's internal processor registers (IPRs)
- Sets up the microprocessor's internal L1/L2 caches
- Performs the minimum I/O subsystem initialization necessary to access the realtime clock (RTC) and the system's flash ROM
- Detects CPU speed by polling the periodic interrupt flag (PIF) in the RTC
- Sets up memory and backup cache (Bcache) parameters based on the speed of the CPU
- Wakes up the DRAMs
- Initializes the Bcache
- Copies the contents of the entire system memory to itself to ensure good memory data parity
- Scans the system flash ROM for a special header that specifies where and how the system flash ROM firmware should be loaded
- Copies the contents of the system flash ROM to memory and begins code execution
- Passes parameters up to the next level of firmware to provide a predictable firmware interface

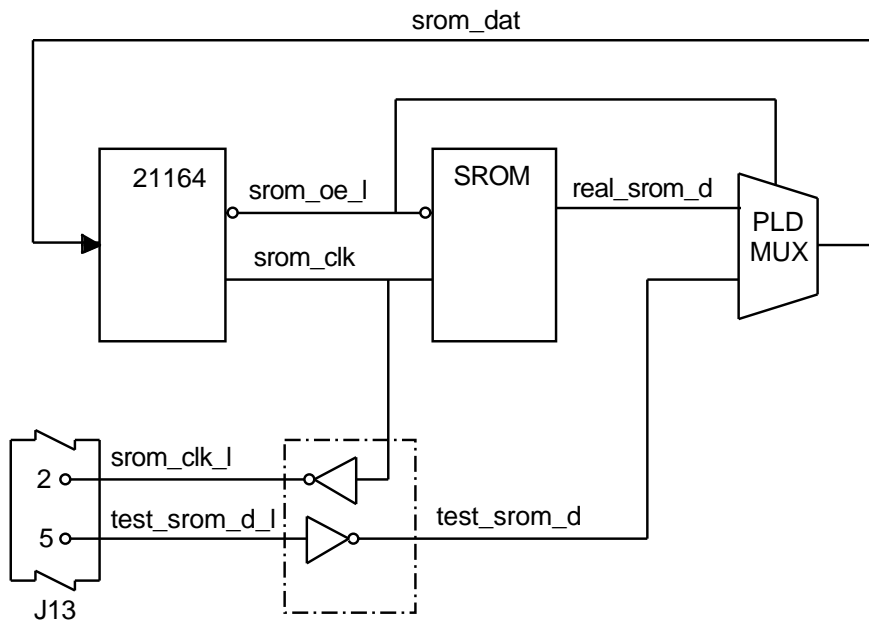
Figure 3-7 is a simplified diagram of the SROM and serial port logic.

Signal **srom\_oe\_1** selects the input to the multiplexer in the MACH210A programmable logic device (PLD). The multiplexer selects either the output of the Xilinx XC17128 SROM (**real\_srom\_d**) or a user-supplied input through the test SROM port (**test\_srom\_d**). The multiplexer output (**srom\_d**) provides data input (**srom\_dat**) to the 21164.

After the initial SROM code has been read into the 21164's Icache, the test SROM port can be used as a software-controlled serial port. This serial port can be used for diagnosing system problems when the only working devices are the microprocessor, the SROM, and the circuits needed for the direct support of the microprocessor and SROM such as the clock. Connector J13 (see Figure 2-3 and Table 2-2) supports an RS232 or RS422 terminal connection to this port by using 1488 and 1489 line driver and receiver components. Additional external logic is not required.

### 3.10 dc Power Distribution

Figure 3-7 SROM and Serial Port



SW-2306-17

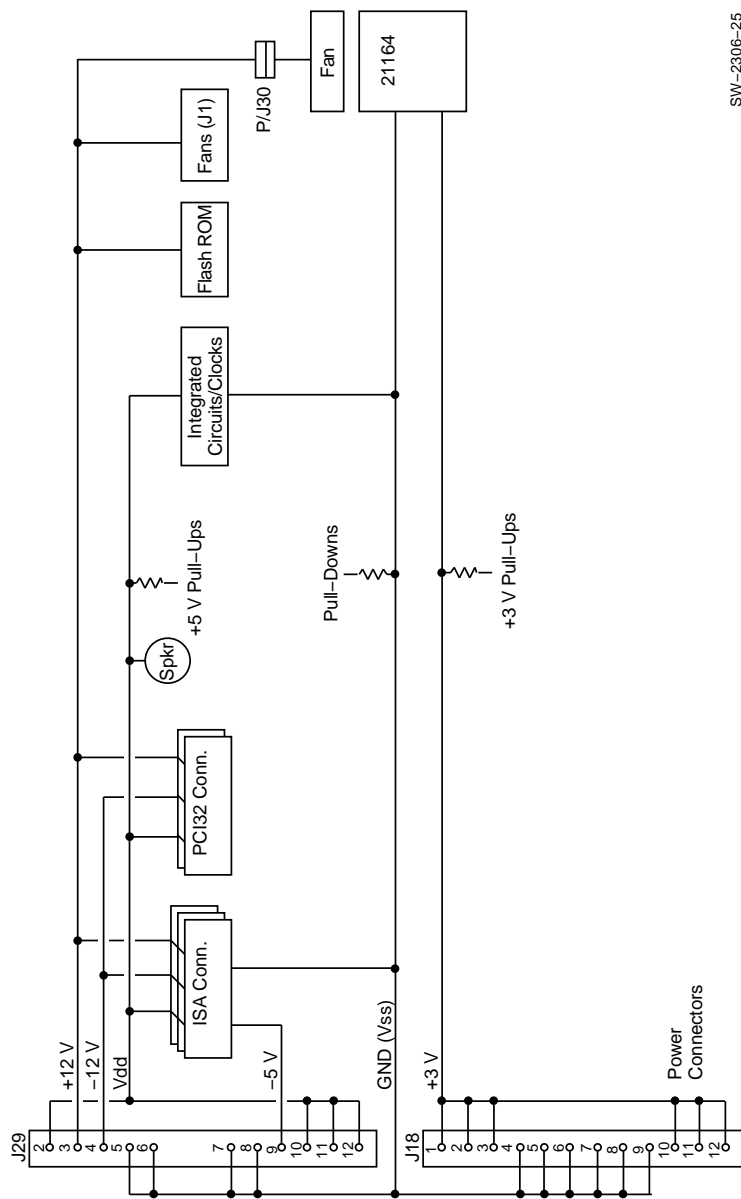
### 3.10 dc Power Distribution

The Alpha PCI 164-266/300 derives its system power from a user-supplied PC power supply. The power supply must provide  $\pm 12$  V dc,  $\pm 5$  V dc, and +3.3 V dc. The dc power is supplied through power connectors J18 and J29. (See Figure 3-8.) Power is distributed to the board logic through dedicated power planes within the 6-layer board structure.

As shown in Figure 3-8, the +12 V dc, -12 V dc, and -5 V dc are supplied to ISA connectors J19, J20, and J21. The +12 V dc and -12 V dc are supplied to ISA connectors and PCI32 connectors J24 and J25. The +12 V dc is also supplied to the CPU fan connector J30, auxiliary fan connector pins on header J2, and to the flash ROM write-enable connector J14. +5 V is supplied to ISA connectors, PCI32 connectors and most of the board's integrated circuits. The +3.3 V dc is supplied to the 21164 microprocessor.

### 3.10 dc Power Distribution

Figure 3-8 dc Power Distribution



SIW-2306-25

## 3.11 System Software

### 3.11 System Software

Alpha PCI 164-266/300 software consists of the following:

- Serial ROM code
- Mini-Debugger code
- Debug monitor ROM code
- Windows NT ARC firmware
- Operating systems

The serial ROM code, Mini-Debugger code, debug monitor code, and Windows NT ARC firmware are all included with the Alpha PCI 164-266/300 and do not require a license. Only binaries for the Windows NT ARC firmware are included, not the sources. Operating systems are available as licensed products. Refer to Appendix C for a list of related documentation.

#### 3.11.1 Serial ROM Code

The serial ROM code is contained in the Xilinx XC17128 serial configuration ROM. This code is executed by the 21164 when system power is turned on (see Section 3.9). The serial ROM code initializes the system, then transfers control to either the Mini-Debugger, the debug monitor, or the selected firmware, depending upon the setting of the configuration jumpers CONF10 and CONF11.

#### 3.11.2 Mini-Debugger Code

The Alpha SROM Mini-Debugger provides basic hardware debugging capability through a serial connector interface to the SROM port of the 21164. Using only an SROM containing this program, a clock source, the Alpha 21164 microprocessor, and a few gates, you can exercise devices connected to the CPU to debug caches, main memory, and I/O subsystems until the board is functional enough to support a more fully featured monitor.

The Mini-Debugger provides:

- Basic hardware debugging capability
- The capability to load code through the SROM test port
- A monitor that can point to hardware addresses and exercise registers and devices at those locations
- The ability to examine and deposit memory locations
- A case-independent command language

## 3.11 System Software

- Support for variable CPU speeds and communication baud rates

For additional information, refer to the *Alpha Microprocessors SROM Mini-Debugger User's Guide*.

### 3.11.3 Debug Monitor ROM Code

The Alpha PCI 164-266/300 includes a flash ROM that contains the debug monitor code. Room is provided in this ROM for user-specific code development. This code can be loaded independently. The user can develop code on a host system, then load the code into the Alpha PCI 164-266/300 board through an Ethernet board or diskette.

The functions provided by the debug monitor include:

- File load
- Read/write memory and registers
- Memory image dump
- Transfer control to program
- Breakpoints
- Single stepping
- Disassembly
- Source-level remote debug

The user kit includes the full source code listing for all SROM and debug monitor ROM software. For additional information, refer to the *Alpha Microprocessors Evaluation Board Debug Monitor User's Guide*.

### 3.11.4 Operating Systems

The Alpha PCI 164-266/300 is designed to run the Windows NT licensed operating system.

# 4

---

## System Address Mapping

This chapter describes the mapping of the 40-bit processor physical address space into cacheable and noncacheable memory addresses, the translation of the processor-initiated address into a peripheral component interconnect (PCI) space address, and the translation of PCI-initiated addresses into system memory addresses.

### 4.1 Physical Memory Regions

The Alpha PCI 164-266/300 physical address space, as seen from the 21164, is divided into three regions:

- Region 1—Cacheable space
- Region 2—Noncacheable space
- Region 3—A region reserved for 21164 cache control and bus interface unit (Cbox) internal processor registers (IPRs)

In region 1, write-back caching, write-merging operations, and load-merging operations are all permitted. In regions 2 and 3, load-merging operations are permitted, but the request includes a mask to tell the system environment which INT8s are being accessed.

Write-merging operations are also permitted, and the mask indicates which INT4s are actually modified. The 21164 does not generate accesses to the third region if they map to a Cbox IPR. Accesses in this region that are not to a defined Cbox IPR produce UNDEFINED results.

All cacheable accesses by the 21164 microprocessor, as implemented on the Alpha PCI 164-266/300 produce 64-byte requests. All noncacheable (I/O space) references produce 32-byte requests. Table 4-1 describes each of the three regions.

## 4.1 Physical Memory Regions

**Table 4–1 Three Physical Memory Regions**

Region	Address Range <sub>16</sub>	Description
Cacheable	00.0000.0000–7F.FFFF.FFFF	Write-back cached, load and store merging operations permitted, 64-byte transfers.
Noncacheable	80.0000.0000–FF.FFEF.FFFF	Not cached, load-merging operations limited, store-merging operations permitted, 32-byte transfers.
Cbox IPR region	FF.FFF0.0000–FF.FFFF.FFFF	Cbox IPRs. Accesses do not appear on the interface pins unless an undefined location is accessed (produces UNDEFINED results).

Table 4–2 provides a more detailed description of the Alpha PCI 164-266/300's address mapping. The noncached space for Alpha PCI 164-266/300 contains the system control and status registers (CSR's), noncached memory access (for diagnostics), and the PCI address space. The PCI defines three physical address spaces: a 4GB PCI memory space, a 4GB PCI I/O space, and a 256-byte-per-device PCI configuration space. The noncached space is also used to generate PCI interrupt acknowledge and special cycles.

The 21164 microprocessor has visibility to the complete address space. However, the PCI devices have a restricted view of the address space. They can access any PCI device through the PCI memory or I/O space, but they have no access to the PCI configuration space. The Alpha PCI 164-266/300 board provides direct memory access (DMA) to the system memory through four programmable windows in the PCI memory space. These address windows are a PCI requirement. Each window is defined by a base register and is implemented by all PCI devices.

DMA access to system memory is achieved by either direct-mapped access or through scatter-gather translation. Direct-mapped accesses are performed by concatenating an offset to a portion of the PCI address. The scatter-gather mappings translate any 8KB PCI memory address region into a corresponding 8KB cached memory region.



## 4.1 Physical Memory Regions

**Table 4–2 Physical Memory Regions (Detailed)**

Region <sub>16</sub>	Description
00.0000.0000–00.3FFF.FFFF	Cacheable memory space (1GB)
00.4000.0000–7F.FFFF.FFFF	UNDEFINED space (511GB)
80.0000.0000–83.FFFF.FFFF	PCI sparse memory space—region 0 (16GB through 512MB)
84.0000.0000–84.FFFF.FFFF	PCI sparse memory space—region 1 (4GB through 128MB)
85.0000.0000–85.7FFF.FFFF	PCI sparse memory space—region 2 (2GB through 64MB)
85.8000.0000–85.BFFF.FFFF	PCI sparse I/O space—region A (1GB through 32MB)
85.C000.0000–85.FFFF.FFFF	PCI sparse I/O space—region B (1GB through 32MB)
86.0000.0000–86.FFFF.FFFF	PCI dense memory space (4GB)
87.0000.0000–87.1FFF.FFFF	PCI configuration space (512MB)
87.2000.0000–87.3FFF.FFFF	PCI interrupt acknowledge/special cycle space (512MB)
87.4000.0000–87.4FFF.FFFF	CIA main CSR space (256MB)
87.5000.0000–87.5FFF.FFFF	CIA memory control CSR space (256MB)
87.6000.0000–87.6FFF.FFFF	CIA PCI address translation (256MB)
87.7000.0000–FF.FFEF.FFFF	UNDEFINED space (~482GB)
FF.FFF0.0000–FF.FFFF.FFFF	Cbox IPR space (1MB)

The Alpha PCI 164-266/300 uses a flush-based cache coherence protocol. All DMA read operation requests are serviced by the 21164 if the data resides in its caches. Otherwise, data will be returned from memory. All DMA write operation requests will invalidate matching addresses in the 21164's caches and, if the cache entry was dirty, will be merged with the DMA write data before being written to memory. One exception to this occurs when a DMA locked read operation is performed; the Alpha PCI 164-266/300 will treat the read operation like a DMA write operation except that the only data that will be written to memory is cache data if it is dirty. This is done to clear the lock flag in the 21164 and to flush out the locked block from its caches. All read misses from the 21164 that subsequently match the locked address will be stalled until the PCI lock is relinquished. This prevents the 21164 from gaining access to the locked block.

## 4.1 Physical Memory Regions

---

### Caution

---

Due to CIA chip pin constraints, CPU address bits <38:35> are not brought onchip. Software must ensure that CPU address bits <38:35> are always zero (to ensure even parity). Otherwise, the CIA chip will generate parity error interrupts during address cycles.

---

## 4.2 Alpha 21164 Address Mapping to PCI Space

The control, I/O interface, and address chip (CIA) generates 32-bit PCI addresses but accepts both 64-bit address (DAC <sup>1</sup>) cycles and 32-bit PCI address (SAC <sup>2</sup>) cycles. However, the Alpha PCI 164-266/300 supports only up to a maximum of 512MB of main memory, which precludes any benefit from the CIA chip accepting 64-bit DAC addressing.

The Alpha PCI 164-266/300 provides 4GB of PCI dense space to map the complete PCI memory space. PCI sparse memory space of 704MB is provided, which has byte granularity and is the safest memory space to use with respect to merging and prefetching. The Alpha PCI 164-266/300 provides three PCI sparse memory regions that can be relocated by means of the HAE\_MEM CSR as follows:

- 512MB region that can be located in any NATURALLY ALIGNED 512MB segment of the PCI memory space. This region might be sufficient for software needs and the remaining two regions could be ignored.
- 128MB regions that can be located on any NATURALLY ALIGNED 128MB segment of the PCI memory space.
- 64MB region that can be located on any NATURALLY ALIGNED 128MB segment of the PCI memory space. This range is intended to be located in the PCI address segment 0–64MB for ISA space accesses. However, the region can be relocated for software convenience.

The Alpha PCI 164-266/300 provides 64MB of PCI sparse I/O space. PCI devices will probably not exceed 64KB in the near future, thus 64MB should provide ample space. The PCI I/O sparse space is divided into two 32MB regions. Region A is fixed in PCI segment 0–32MB. Region B can be relocated to any 32MB region through use of the HAE\_IO register.

---

<sup>1</sup> Dual-address cycle—used only if address bits <63:32> are non-zero.

<sup>2</sup> Single-address cycle—used for 32-bit PCI addresses, or if bits <63:32> are zero for a 64-bit address.

## 4.2 Alpha 21164 Address Mapping to PCI Space

### 4.2.1 Cacheable Memory Space (00.0000.0000 Through 00.3FFF.FFFF)

The Alpha PCI 164-266/300 recognizes the first 1GB of the physical address space to be cacheable memory space. It responds to all read and write accesses in this space. The block size is fixed at 64 bytes.

The Alpha PCI 164-266/300 uses a read/flush-based cache coherence protocol. All DMA read accesses are sent to the 21164 as read probes while all DMA write accesses are sent to the 21164 as flush probes. DMA read operations that hit in any of the caches will cause data to be returned from that cache. DMA read operations that miss all of the caches cause data to be returned from memory. DMA write operations that hit in any of the caches will cause the entry to be flushed (invalidated), and if dirty, the data will be merged with the DMA write data before being written to memory.

### 4.2.2 PCI Sparse Memory Space (80.0000.0000 Through 85.7FFF.FFFF)

The Alpha PCI 164-266/300 provides three regions of contiguous CPU address space that maps to PCI sparse memory space. Accesses to this space can have byte, word, tribyte, longword, or quadword granularity, which the PCI requires, even though the Alpha architecture does not provide byte, word, or tribyte granularity. In addition, Intel processors are capable of generating UNALIGNED references, and it is desirable to emulate the resulting PCI transactions to ensure compatibility with PCI devices designed for Intel platforms.

Therefore, to provide this granularity, the byte enable and byte length information is encoded in the lower address bits in this space. Address bits <6:3> are used for this purpose. Bits <31:7> are used to generate longword addresses on the PCI bus, thus resulting in a sparse 4GB space that maps to 128MB of address space on the PCI. An access to this space causes a memory read or memory write access on the PCI.

The rules for accessing this region are as follows:

- Sparse space supports all the byte encodings that can be generated in an Intel platform to ensure compatibility with PCI device drivers. The results of some references are not explicitly defined. The Alpha PCI 164-266/300 completes the reference UNPREDICTABLY but does not report an error.<sup>3</sup>

---

<sup>3</sup> The CIA chip generates an interrupt if it is enabled in the CIA\_ERROR\_MASK register.

## 4.2 Alpha 21164 Address Mapping to PCI Space

- Software must use longword load or store instructions (LDL/STL) to perform a reference that is of longword length (or less) on the PCI. The bytes to be transferred must be positioned within the longword in the correct byte lanes as indicated by the PCI byte enables. The Alpha PCI 164-266/300 performs no byte shifting within the longword. Quadword load and store instructions must only be used to perform a quadword transfer. Use of STQ/LDQ instructions for any other references will produce UNPREDICTABLE results.
- The Alpha PCI 164-266/300 does not prefetch in sparse space (no side effects).
- Accesses in this space are no greater than one quadword. Software must ensure that the processor does not merge consecutive read or write transactions by using memory barrier (MB) instructions after each read or write transaction in this address space. However, consecutive sparse space addresses (that is, to a different PCI longword/quadword) will be separated by at least 32 bytes and are not merged by the Alpha PCI 164-266/300.
- Software must insert MB instructions if the sparse space address can alias to a dense space address. Otherwise, ordering and coherency cannot be maintained.
- The encoding of the 21164 address for sparse space read accesses to PCI space is shown in Table 4-3. It is important to note that CPU address bits <33:5> are directly available from the 21164 pins. On read operations, address bits <4:3> can be calculated from the INT4\_VALID pins. CPU address bits <2:0> are required to be zero.
- The relationship between INT4\_VALID and CPU address bits <4:3> for sparse space write operations is shown in Table 4-4.

Table 4-3 defines the low-order PCI sparse memory address bytes. CPU address bits <7:3> are used to generate the length of the PCI transaction in bytes, the byte-enables, and address bits <2:0>. CPU address bits <30:8> correspond to the quadword PCI address and are sent out on PCI address <25:3>.

## 4.2 Alpha 21164 Address Mapping to PCI Space

**Table 4–3 PCI Sparse Memory Space Byte-Enable Generation**

Length <sup>1</sup>	CPU Address <6:5>	CPU Address <4:3>	PCI Byte-Enable <sup>2</sup>	PCI Address <2:0> <sup>3</sup>
Byte	00	00	1110	<b>pci_ad&lt;7&gt;</b> , 00
	01	00	1101	<b>pci_ad&lt;7&gt;</b> , 00
	10	00	1011	<b>pci_ad&lt;7&gt;</b> , 00
	11	00	0111	<b>pci_ad&lt;7&gt;</b> , 00
Word	00	01	1100	<b>pci_ad&lt;7&gt;</b> , 00
	01	01	1001	<b>pci_ad&lt;7&gt;</b> , 00
	10	01	0011	<b>pci_ad&lt;7&gt;</b> , 00
Tribyte	00	10	1000	<b>pci_ad&lt;7&gt;</b> , 00
	01	10	0001	<b>pci_ad&lt;7&gt;</b> , 00
Longword	00	11	0000	<b>pci_ad&lt;7&gt;</b> , 00
Quadword	11	11	0000	000

<sup>1</sup>Missing entries have UNPREDICTABLE results.

<sup>2</sup>Byte-enable set to 0 indicates that byte lane carries meaningful data.

<sup>3</sup>In PCI sparse memory space, PCI address bits <1:0> are always 00.

**Table 4–4 INT4\_VALID to Address Translation for Sparse Write Operations**

21164 Data Cycle <sup>1</sup>	int4_valid<3:0>	CPU Address <4:3>
First	0001	00
First	0010	00
First	0100	01
First	1000	01
Second	0001	10
Second	0010	10
Second	0100	11
Second	1000	11
Second	1100 (STQ) <sup>2</sup>	11

<sup>1</sup>Missing entries have UNPREDICTABLE results.

<sup>2</sup>Only one valid STQ case is allowed.

## 4.2 Alpha 21164 Address Mapping to PCI Space

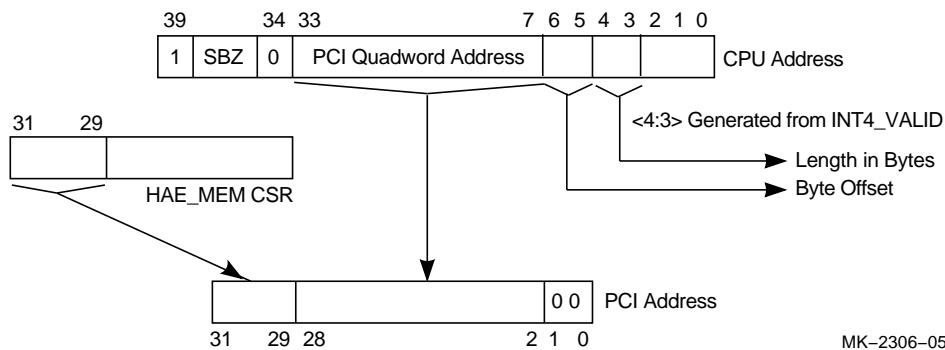
PCI address bits <31:26> are obtained from either the hardware extension register (HAE\_MEM), or the CPU address, depending upon the sparse space being accessed. This is shown in Table 4–5. HAE\_MEM is a CSR in the CIA chip and is described in Section A.6.1.

**Table 4–5 High-Order Sparse Space Bits**

PCI Address Bit	Region 1	Region 2	Region 3
<b>PCI_AD&lt;31&gt;</b>	HAE_MEM<31>	HAE_MEM<15>	HAE_MEM<7>
<b>PCI_AD&lt;30&gt;</b>	HAE_MEM<30>	HAE_MEM<14>	HAE_MEM<6>
<b>PCI_AD&lt;29&gt;</b>	HAE_MEM<29>	HAE_MEM<13>	HAE_MEM<5>
<b>PCI_AD&lt;28&gt;</b>	<b>addr&lt;33&gt;</b>	HAE_MEM<12>	HAE_MEM<4>
<b>PCI_AD&lt;27&gt;</b>	<b>addr&lt;32&gt;</b>	HAE_MEM<11>	HAE_MEM<3>
<b>PCI_AD&lt;26&gt;</b>	<b>addr&lt;31&gt;</b>	<b>addr&lt;31&gt;</b>	HAE_MEM<2>

Figure 4–1, Figure 4–2, and Figure 4–3 illustrate the PCI sparse memory space translation for each region.

**Figure 4–1 PCI Sparse Memory Space Address Translation—Region 1**



MK–2306–05

## 4.2 Alpha 21164 Address Mapping to PCI Space

Figure 4-2 PCI Sparse Memory Space Address Translation—Region 2

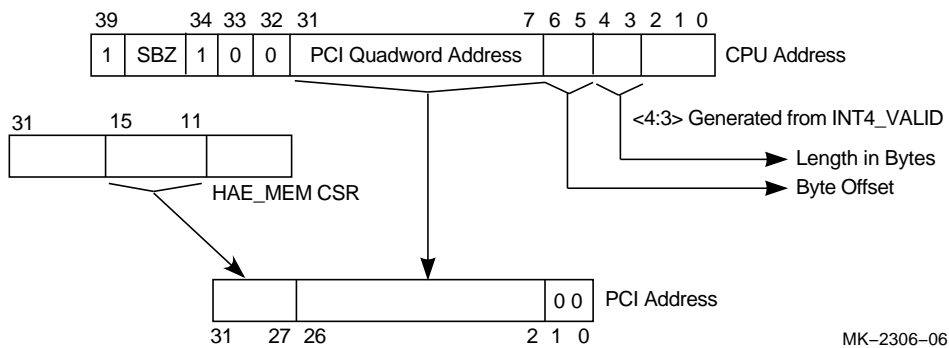
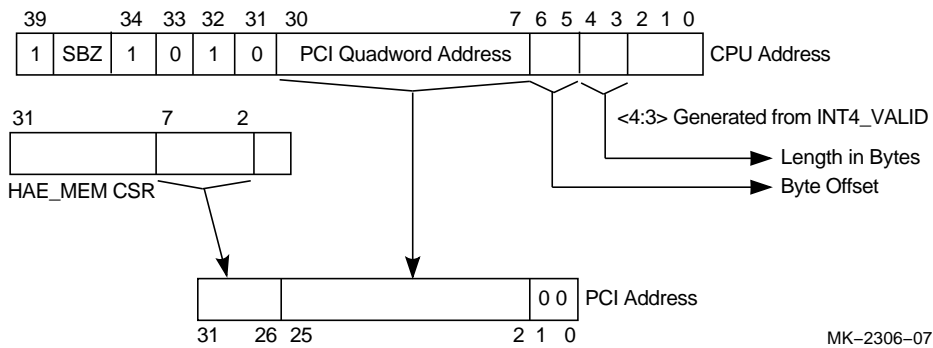


Figure 4-3 PCI Sparse Memory Space Address Translation—Region 3



### 4.2.3 Hardware Extension Registers (HAE)

In sparse space, CPU address bits <7:3> are wasted on encoding byte-enables, size, and the low-order PCI address bit <2>. Therefore, there are five fewer address bits available to generate the PCI physical address. Hardware extension registers (HAEs) are used to provide the missing high-order bits. The HAE registers are expected to be set by firmware.

## 4.2 Alpha 21164 Address Mapping to PCI Space

The Alpha PCI 164-266/300 provides three sparse space PCI memory regions and allows each to be relocated by means of bits in the HAE\_MEM register located in the CIA chip. Two regions of PCI I/O sparse space are provided, region A and region B. Region A addresses the lower 32MB of PCI I/O space and is never relocated. This region is intended to be used to address ISA devices. Region B is used to address a 32MB region that can be relocated by using the HAE\_IO register located in the CIA chip.

### 4.2.4 PCI Sparse I/O Space (85.8000.0000 Through 85.FFFF.FFFF)

PCI sparse I/O space is sparse and has characteristics similar to the PCI sparse memory space. This 2GB physical address space maps to two 32MB regions of PCI I/O address space. A read or write operation to this space causes a PCI I/O read or PCI I/O write command, respectively.

The address generation is as follows:

- Region A: This region has CPU address bits <34:30> = 10110 and addresses the lower 32MB of PCI sparse I/O space. Therefore, PCI address bits <31:25> are set to zero by the hardware. This region is used for ISA addressing.
- Region B: This region has CPU address bits <34:30> = 10111 and addresses 32MB of PCI sparse I/O space that can be relocated. This 32MB segment is relocated by assigning HAE\_IO<31:25> to PCI address<31:25>.
- PCI address bits <24:3> are derived from CPU address bits <29:8>.
- PCI address bits <2:0> are defined in Table 4-6.

The lower 64KB of PCI sparse I/O space should be reserved for the ISA devices. Therefore, all PCI devices should be relocated above this region.



## 4.2 Alpha 21164 Address Mapping to PCI Space

**Table 4–6 PCI Sparse I/O Space Byte-Enable Generation**

Length <sup>1</sup>	CPU Address <6:5>	CPU Address <4:3>	PCI Byte-Enable <sup>2</sup>	PCI Address <2:0>
Byte	00	00	1110	<b>pci_ad&lt;7&gt;, 00</b>
	01	00	1101	<b>pci_ad&lt;7&gt;, 01</b>
	10	00	1011	<b>pci_ad&lt;7&gt;, 10</b>
	11	00	0111	<b>pci_ad&lt;7&gt;, 11</b>
Word	00	01	1100	<b>pci_ad&lt;7&gt;, 00</b>
	01	01	1001	<b>pci_ad&lt;7&gt;, 01</b>
	10	01	0011	<b>pci_ad&lt;7&gt;, 10</b>
Tribyte	00	10	1000	<b>pci_ad&lt;7&gt;, 00</b>
	01	10	0001	<b>pci_ad&lt;7&gt;, 01</b>
Longword	00	11	0000	<b>pci_ad&lt;7&gt;, 00</b>
Quadword	11	11	0000	000

<sup>1</sup>Missing entries produce UNPREDICTABLE results.

<sup>2</sup>Byte-enable set to 0 indicates that byte lane carries meaningful data.

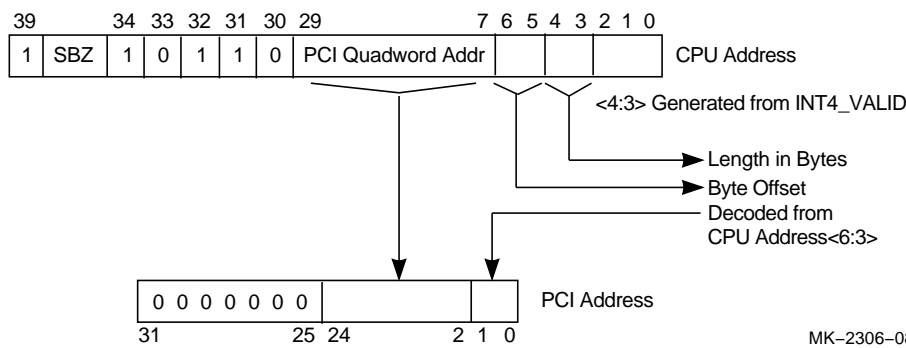
### Caution

Quadword accesses to PCI sparse I/O space will cause a two-longword burst on the PCI bus. Some PCI devices might not support bursting in I/O space.

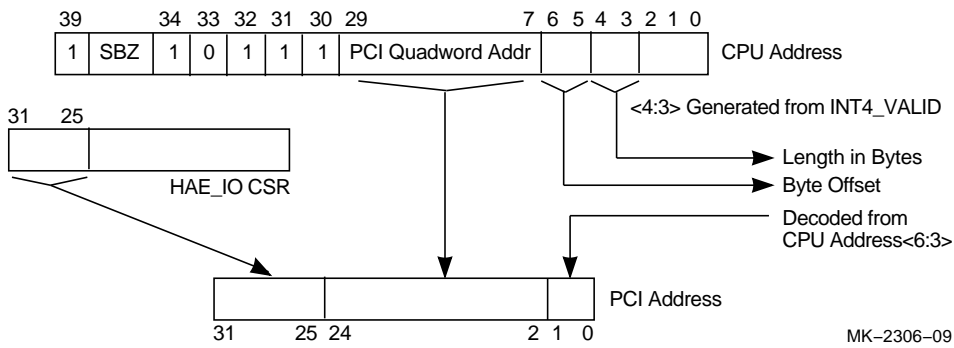
Figure 4–4 and Figure 4–5 illustrate the PCI sparse I/O space translation for each region.

## 4.2 Alpha 21164 Address Mapping to PCI Space

**Figure 4-4 PCI Sparse I/O Space Address Translation—Region A**



**Figure 4-5 PCI Sparse I/O Space Address Translation—Region B**



## 4.2 Alpha 21164 Address Mapping to PCI Space

### 4.2.5 PCI Dense Memory Space (86.0000.0000 Through 86.FFFF.FFFF)

PCI dense memory space is typically used for PCI data buffers (such as a video frame buffer) and has the following characteristics:

- There is one-to-one mapping between CPU addresses and PCI addresses. A longword address from the CPU maps to a longword address on the PCI. Hence, the name dense space (as opposed to PCI sparse memory space).
- Byte or word accesses are not permitted in this space. Minimum access granularity is a longword on write operations and a quadword on read operations. The maximum transfer length is 32 bytes on write operations (performed as a burst of 8 longwords on the PCI) and on read operations. Any combination of longwords may be valid on write operations. Valid longwords surrounding an invalid longword (or longwords), called a hole, are required to be handled correctly by all PCI devices.
- The 21164 cannot specify a longword address for read transactions. The minimum granularity it can specify is a quadword in noncacheable space. Therefore, minimum granularity read operations in this space should always be executed as a quadword read operation with a burst length of two on the PCI bus. The 21164 merges noncached read operations up to a 32-byte maximum. The largest dense space read operation is therefore 32 bytes. This space cannot be used for devices that have read side effects due to load-merging and prefetching operations. Any prefetched data is not cached between transactions. The burst read on the PCI bus is not atomic.
- Write operations to addresses in this space can be buffered in the 21164. The Alpha PCI 164-266/300 supports a burst length of eight on the PCI, corresponding to 32 bytes of data. In addition, the CIA chip provides four 32-byte write buffers to increase I/O write performance. These four buffers are strictly ordered.

The address generation in dense space is as follows:

- CPU address bits <31:5> are directly sent out on PCI address<31:5>.
- CPU address bits <4:3> are generated from the 21164's **int4\_valid\_h** signals.
- On read transactions, PCI address bit <2> is always 0.
- On write transactions, PCI address bit <2> is generated from the **int4\_valid\_h** signals. If the lower longword is to be written, PCI address bit <2> = 0. If the lower longword is masked out and the upper longword is to be written, PCI address bit <2> = 1.
- PCI address bits <1:0> are forced to 0.

## 4.2 Alpha 21164 Address Mapping to PCI Space

---

### Note

---

If the data written by the processor has holes, that is, some of the longwords have been masked out, the corresponding transfer will still be performed on the PCI bus with disabled byte-enables. Downstream bridges must be able to deal with disabled byte-enables on the PCI bus during write transactions.

---

PCI dense memory space has the following advantages over PCI sparse memory space:

- Some software requires memory-like accesses such that accesses on the PCI are adjacent Alpha addresses.
- PCI bus burst transfers are not possible in sparse space apart from two longword bursts for quadword write operations. Dense space allows both read and write bursting.
- Dense space allows separate accesses to be merged in read and write buffers. This is not allowed in sparse space.
- In general, sparse space accesses are separated by memory barriers to avoid read/write buffer merging. Dense space accesses only require memory barriers when explicit ordering is required by software. Therefore, fewer MB instructions should be needed.

### 4.2.6 PCI Configuration Space (87.0000.0000 Through 87.1FFF.FFFF)

A read or write access to the PCI configuration space causes a configuration read or write cycle on the PCI. There are two classes of targets, which are selected based on the value of the CIA chip's PCI configuration register (CFG):

- Type 0: These are targets on the Alpha PCI 164-266/300's primary 64-bit PCI bus. These are selected when CFG<1:0> = 00.
- Type 1: These are targets on a secondary PCI bus. These are selected when CFG<1:0> = 01.

Software must first program the CFG register before running a configuration cycle. Sparse address decoding is used. CPU address bits <6:3> are used to generate both the length of the PCI transaction in bytes, and the byte-enables. PCI address bits <1:0> are obtained from CFG<1:0>. CPU address bits <28:7> correspond to PCI address bits <23:2> and provide the configuration command information. The high-order PCI address bits <31:24> are always zero.

## 4.2 Alpha 21164 Address Mapping to PCI Space

There are two classes of targets for PCI configuration read and write commands: devices on the primary PCI bus and peripherals on hierarchical (buffered, secondary) PCI buses, which are accessed through bridge chips. Address usage during PCI configuration cycles varies depending on the intended target of the configuration cycle.

Peripherals are selected during a PCI configuration cycle if their initialization device select (IDSEL) pin is asserted, the PCI bus command indicates a configuration read or write operation, and address bits <1:0> are 00 (type 0). Address bits <7:2> select a dword (longword) register in the peripheral's 256-byte configuration address space. Accesses can use byte masks. Peripherals that integrate multiple functional units (example: SCSI and Ethernet) can provide configuration spaces for each function. Address bits <10:8> can be decoded by the peripheral to select one of eight functional units. Address bits <31:11> are used to generate IDSEL signals. Typically, the IDSEL pin of each PCI peripheral is connected to a unique address line. This requires that only one bit of the field AD<31:11> is asserted in a given cycle. The Alpha PCI 164-266/300 forces zeros on PCI\_AD<31:24> during configuration cycles. Therefore, only PCI\_AD<23:11> can be used to drive an IDSEL.

If the PCI cycle is a configuration read or write cycle, but address bits <1:0> = 01 (type 1), then a device on a hierarchical bus is being selected through a PCI/PCI bridge chip. This cycle will be accepted by a PCI/PCI bridge for propagation to its secondary PCI interface. During this cycle, bits AD<23:16> select a unique bus number, bits AD<15:8> select a device on that bus (typically decoded by the target bridge to generate an IDSEL), and bits AD<7:2> select a dword (longword) in the device's configuration register space.

Each PCI/PCI bridge device can be configured by PCI configuration cycles through the primary PCI interface. Configuration parameters in the PCI/PCI bridge identify the bus number for its secondary PCI interface and a range of bus numbers that may exist hierarchically behind it.

If the bus number of the configuration cycle matches the bus number of the bridge chip's secondary PCI interface, it will intercept the configuration cycle, decode it, and generate a PCI configuration cycle with AD<1:0> = 00 on its secondary PCI interface. If the bus number is within the range of bus numbers that can exist hierarchically behind its secondary PCI interface, the bridge passes the PCI configuration cycle on unmodified (AD<1:0> = 01). It will be intercepted and decoded by a downstream bridge.

Table 4-7 defines the various PCI\_AD fields during the address phase of an Alpha PCI 164-266/300 configuration read/write cycle.

## 4.2 Alpha 21164 Address Mapping to PCI Space

**Table 4–7 PCI Configuration Space Definition**

Bus Hierarchy	PCI_AD Bits	Definition
Local	<31:24>	Forced to 0 by the CIA chip.
	<23:11>	Can be used for IDSEL or ignored. Typically, the IDSEL pin of each PCI device is connected to a different address line. This requires that only one bit of this field be asserted in a given cycle.
	<10:8>	Function select (1 of 8).
	<7:2>	Register select.
	<1:0>	00
Remote <sup>1</sup>	<31:24>	Ignored—forced to 0 by the CIA chip.
	<23:16>	Bus number.
	<15:11>	Device number.
	<10:8>	Function select (1 of 8).
	<7:2>	Register select.
<1:0>	01	

<sup>1</sup>Communication is through a PCI/PCI bridge.

Table 4–8 defines the PCI configuration space byte-enable field encoding.

## 4.2 Alpha 21164 Address Mapping to PCI Space

**Table 4–8 PCI Configuration Space Byte-Enable Generation**

Length <sup>1</sup>	CPU addr_h<6:5>	CPU addr_h<4:3>	PCI Byte-Enable <sup>2</sup>	pci_ad<1:0>
Byte	00	00	1110	CFG<1:0>
	01	00	1101	CFG<1:0>
	10	00	1011	CFG<1:0>
	11	00	0111	CFG<1:0>
Word	00	01	1100	CFG<1:0>
	01	01	1001	CFG<1:0>
	10	01	0011	CFG<1:0>
Tribyte	00	10	1000	CFG<1:0>
	01	10	0001	CFG<1:0>
Longword	00	11	0000	CFG<1:0>
Quadword	11	11	0000	CFG<1:0>

<sup>1</sup>Missing entries produce UNPREDICTABLE results.

<sup>2</sup>Byte-enable set to 0 indicates that byte lane carries meaningful data.

### Caution

If a quadword access is specified for the configuration cycle, then the least significant bit (LSB) of the register number field (PCI\_AD<2>) must be zero (MBZ). Quadword accesses must access quadword-aligned registers.

Table 4–9 shows the Alpha PCI 164-266/300 CPU address encoding for PCI device selection.

## 4.2 Alpha 21164 Address Mapping to PCI Space

Table 4–9 CPU Address Encoding for PCI Device Selection

CPU Address <20:16>	PCI Address Bit	IDSEL
00000	<b>pci_ad&lt;11&gt;</b>	11
00001	<b>pci_ad&lt;12&gt;</b>	12
00010	<b>pci_ad&lt;13&gt;</b>	13
00011	<b>pci_ad&lt;14&gt;</b>	14
00100	<b>pci_ad&lt;15&gt;</b>	15
00101	<b>pci_ad&lt;16&gt;</b>	16
00110	<b>pci_ad&lt;17&gt;</b>	17
00111	<b>pci_ad&lt;18&gt;</b>	18
01000	<b>pci_ad&lt;19&gt;</b>	19
01001	<b>pci_ad&lt;20&gt;</b>	20
01010	<b>pci_ad&lt;21&gt;</b>	21
01011	<b>pci_ad&lt;22&gt;</b>	22
01100	<b>pci_ad&lt;23&gt;</b>	23
01101	<b>pci_ad&lt;24&gt;</b>	24
01110	<b>pci_ad&lt;25&gt;</b>	25
01111	<b>pci_ad&lt;26&gt;</b>	26
10000	<b>pci_ad&lt;27&gt;</b>	27
10001	<b>pci_ad&lt;28&gt;</b>	28
10010	<b>pci_ad&lt;29&gt;</b>	29
10011	<b>pci_ad&lt;30&gt;</b>	30
10100	<b>pci_ad&lt;31&gt;</b>	31
10101 through 11111	No device selection.	

Table 4–10 defines the primary IDSEL mapping for the Alpha PCI 164-266 /300.



## 4.2 Alpha 21164 Address Mapping to PCI Space

**Table 4–10 Alpha PCI 164-266/300 Primary PCI IDSEL Mapping**

IDSEL Device	PCI Address Bit	Physical Address
PCI slot 2	<b>pci_ad&lt;16&gt;</b>	87.0005.0000
PCI slot 0	<b>pci_ad&lt;17&gt;</b>	87.0006.0000
PCI slot 1	<b>pci_ad&lt;18&gt;</b>	87.0007.0000
PCI-to-ISA SIO bridge	<b>pci_ad&lt;19&gt;</b>	87.0008.0000
PCI slot 3	<b>pci_ad&lt;20&gt;</b>	87.0009.0000

### 4.2.7 PCI Interrupt Acknowledge/Special Cycle Space (87.2000.0000 Through 87.3FFF.FFFF)

The special cycle command provides a simple message broadcasting mechanism on the PCI. In general, it can be used for logical sideband signaling between PCI agents.

The special cycle command contains no explicit destination address, but is broadcast to all agents. The Alpha PCI 164-266/300 drives all zeros as the special cycle address. Each receiving agent must determine if the message contained in the data field is applicable to it.

A write access to the CIA chip's IAC\_SC CSR causes a special cycle on the PCI. The 21164's write data is passed unmodified to the PCI. Software must write the data in longword zero of the hexword with the following field:

- Bytes 0 and 1 contain the encoded message.
- Bytes 2 and 3 are message dependent (optional).

A read of the IACK\_SC CSR results in an interrupt acknowledge cycle on the PCI and the return data will be the interrupt vector.

### 4.2.8 Alpha PCI 164-266/300 Hardware-Specific and Miscellaneous Register Space (87.4000.0000 Through 87.6FFF.FFFF)

This address space is a hardware-specific variant of the sparse space encoding. CPU address bits <27:6> specify a longword address where CPU address<5:0> must be zero. All the CIA chip registers are accessed with a longword granularity. Table 4–11 lists each region and the associated addresses. For more specific details on the CIA chip's CSRs, refer to Section A.6 and the *DECchip 21171 Core Logic Chipset Technical Reference Manual*.

## 4.2 Alpha 21164 Address Mapping to PCI Space

**Table 4–11 Hardware-Specific Register Space**

CPU Address <39:28>	Selected Region	CPU Address <27:6>	CPU Address <5:0>
1000 0111 0100	CIA control, diagnostic, error registers	LW address	00 0000
1000 0111 0101	CIA memory control registers	LW address	00 0000
1000 0111 0110	CIA PCI address translation	LW address	00 0000
1000 0111 0111	Reserved	—	—
1000 0111 1xxx	Reserved	—	—

### 4.2.9 PCI-to-Physical Memory Addressing

Incoming 32-bit or 64-bit PCI memory addresses have to be mapped to the Alpha PCI 164-266/300's 30-bit physical memory addresses. The Alpha PCI 164-266/300 provides four programmable address windows that control access of PCI peripherals to system memory. The mapping from the PCI address to the physical address can be direct (physical mapping, with an extension register) or scatter-gather mapped (virtual). These four address windows are referred to as the PCI target windows. The following three registers are associated with each window:

- PCI base register
- PCI mask register
- Translated base register

In addition, there is an extra register associated with window three only. This is the PCI DAC\_BASE register and is used for PCI 64-bit addressing. However, the Alpha PCI 164-266/300 supports only up to a maximum of 512MB of main memory. This precludes any benefit from the CIA chip accepting 64-bit DAC addressing. Refer to the *DECchip 21171 Core Logic Chipset Technical Reference Manual* for more details on DAC addressing (dual address cycle mode).

The PCI mask register provides a mask corresponding to bits <31:20> of an incoming PCI address. The size of each window can be programmed to be from 1MB to 4GB, in powers of two, by masking bits of the incoming PCI address by using the PCI mask register. This is shown in Table 4–12.

## 4.2 Alpha 21164 Address Mapping to PCI Space

**Table 4–12 PCI Target Window Enables**

PCI_MASK<31:20> <sup>1</sup>	Size of Window <sup>2</sup>	Value of $n$ <sup>3</sup>
0000 0000 0000	1MB	20
0000 0000 0001	2MB	21
0000 0000 0011	4MB	22
0000 0000 0111	8MB	23
0000 0000 1111	16MB	24
0000 0001 1111	32MB	25
0000 0011 1111	64MB	26
0000 0111 1111	128MB	27
0000 1111 1111	256MB	28
0001 1111 1111	512MB	29
0011 1111 1111	1GB	30
0111 1111 1111	2GB <sup>4</sup>	31
1111 1111 1111	4GB <sup>4</sup>	32

<sup>1</sup>Combinations of bits in PCI\_MASK<31:20> not shown in this table are not supported and produce UNPREDICTABLE results.

<sup>2</sup>Windows are not allowed to overlap.

<sup>3</sup>Depending upon the target window size, only the incoming address bits <31: $n$ > are compared with bits <31: $n$ > of the PCI base registers as shown in Figure 4–6 ( $n = 20$  to 32). If  $n=32$ , no comparison is performed.  $n$  is also used in Figure 4–7.

<sup>4</sup>This size is not supported by the Alpha PCI 164-266/300.

Based on the value of the PCI mask register, the unmasked bits of the incoming PCI address are compared with the corresponding bits of each window's PCI base register. If the base register and the incoming PCI address match, the incoming PCI address has hit in that PCI target window; otherwise, it has missed in that window. A window enable bit (WENB) is provided in each window's PCI base register to allow windows to be independently enabled or disabled. If a window's WENB bit is set, the window is enabled. The PCI target windows must be programmed so that the PCI address ranges that each one responds to do not overlap; otherwise, the results will be UNDEFINED. The compare scheme between the incoming PCI address and the PCI base register (along with the PCI mask register) previously described, is shown in Figure 4–6.

## 4.2 Alpha 21164 Address Mapping to PCI Space

---

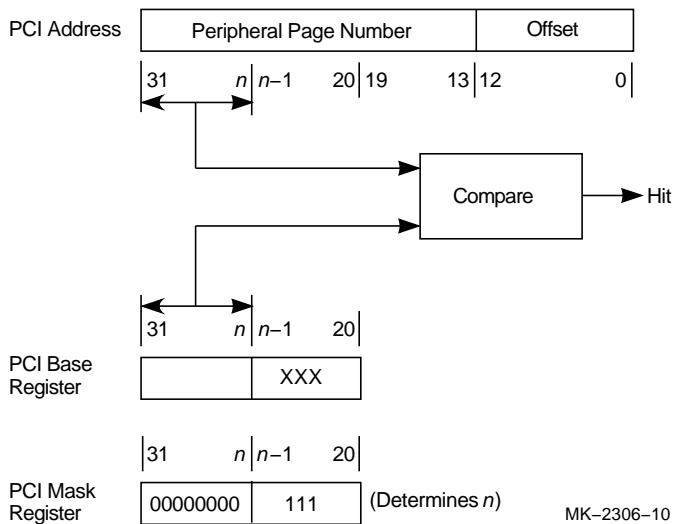
**Note**

---

The window base addresses should be on NATURALLY ALIGNED address boundaries, depending on the size of the window.

---

**Figure 4-6 Addressing Diagram: PCI Target Window Compare**



When an address match occurs with a PCI target window, the Alpha PCI 164-266/300 translates the 32-bit PCI address to a 34-bit processor byte address. The Alpha PCI 164-266/300 supports only a 30-bit memory address. The translated address is generated in one of two ways as determined by the scatter-gather (SG) bit of the window's PCI base register.

If the SG bit is cleared, the DMA address is direct mapped, and the translated address is generated by concatenating bits from the matching window's translated base register with bits from the incoming PCI address. The PCI mask register determines which bits of the translated base register and PCI address are used to generate the translated address as shown in Table 4-13. The unused bits of the translated base register must be cleared for proper operation. Because system memory is located in the lower 1GB of the CPU

## 4.2 Alpha 21164 Address Mapping to PCI Space

address space, address bits <39:33> are always 0. Address <32:5> is obtained from the translated base register.

**Table 4–13 PCI Target Address Translation—Direct Mapped (SG Mapping Disabled)**

PCI_MASK<31:20>	Translated Address <32:5>
0000 0000 0000	T_BASE<32:20>:pci_ad<19:5>
0000 0000 0001	T_BASE<32:21>:pci_ad<20:5>
0000 0000 0011	T_BASE<32:22>:pci_ad<21:5>
0000 0000 0111	T_BASE<32:23>:pci_ad<22:5>
0000 0000 1111	T_BASE<32:24>:pci_ad<23:5>
0000 0001 1111	T_BASE<32:25>:pci_ad<24:5>
0000 0011 1111	T_BASE<32:26>:pci_ad<25:5>
0000 0111 1111	T_BASE<32:27>:pci_ad<26:5>
0000 1111 1111	T_BASE<32:28>:pci_ad<27:5>
0001 1111 1111	T_BASE<32:29>:pci_ad<28:5>
0011 1111 1111	T_BASE<32:30>:pci_ad<29:5>
0111 1111 1111	T_BASE<32:31>:pci_ad<30:5>
1111 1111 1111	T_BASE<32>:pci_ad<31:5>

If the SG 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 table is referred to as a scatter-gather map. The translated base register specifies the starting address of the scatter-gather 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.

Each scatter-gather map entry maps an 8KB page of PCI address space into an 8KB page of the processor's address space. Each scatter-gather map entry is a quadword. Each entry has a valid bit in bit position 0. Address bit <13> is at bit position 1 of the map entry. Because the Alpha PCI 164-266/300 implements valid memory addresses only up to 1GB, bits <63:18> of the scatter-gather map entry should be programmed to 0. Bits <17:1> of the scatter-gather entry are used to generate the physical page address. This is appended to the bits <12:5> of the incoming PCI address to generate the memory address that needs to go out on the system bus.

## 4.2 Alpha 21164 Address Mapping to PCI Space

The size of the scatter-gather map table is determined by the size of the PCI target window as defined by the PCI mask register (Table 4–14). The number of entries is the window size divided by the 8KB page size. The size of the table is the number of entries multiplied by 8 bytes.

**Table 4–14 Scatter-Gather Map Address**

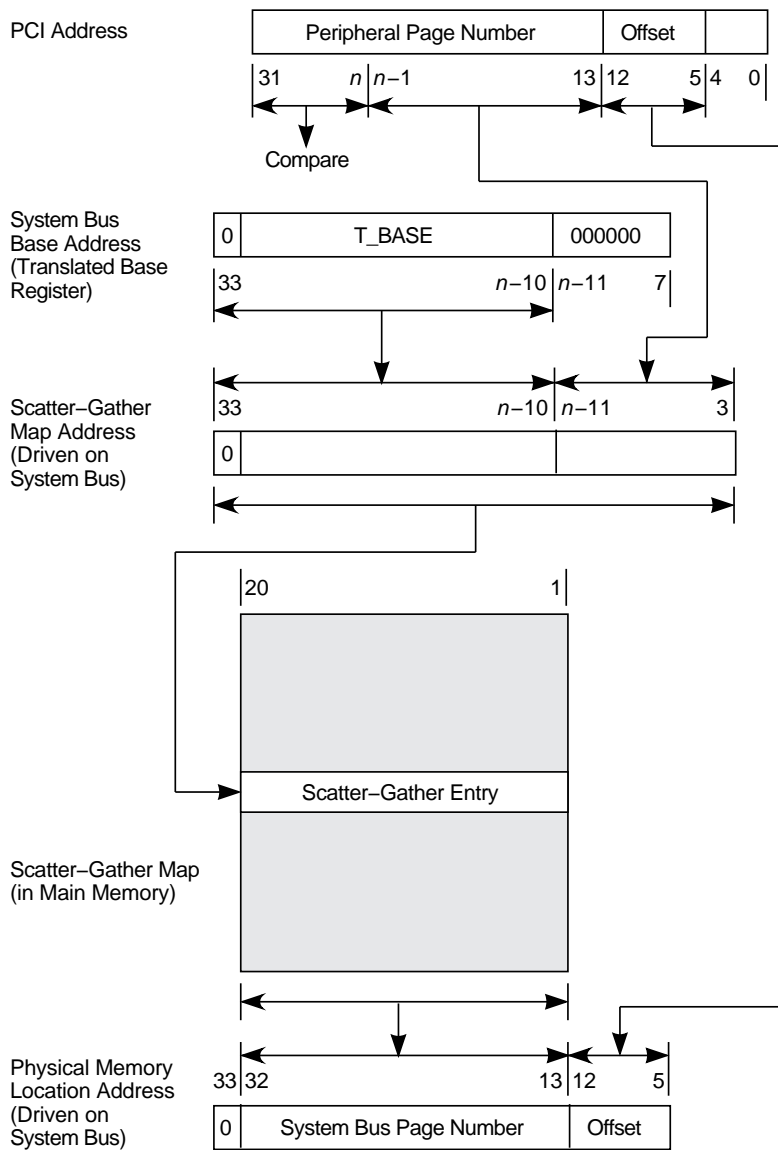
PCI_MASK<31:20>	Size of Window	Scatter-Gather Map Table Size	Scatter-Gather Map Address<32:3>
0000 0000 0000	1MB	1KB	T_BASE<32:10>:pci_ad<19:13>
0000 0000 0001	2MB	2KB	T_BASE<32:11>:pci_ad<20:13>
0000 0000 0011	4MB	4KB	T_BASE<32:12>:pci_ad<21:13>
0000 0000 0111	8MB	8KB	T_BASE<32:13>:pci_ad<22:13>
0000 0000 1111	16MB	16KB	T_BASE<32:14>:pci_ad<23:13>
0000 0001 1111	32MB	32KB	T_BASE<32:15>:pci_ad<24:13>
0000 0011 1111	64MB	64KB	T_BASE<32:16>:pci_ad<25:13>
0000 0111 1111	128MB	128KB	T_BASE<32:17>:pci_ad<26:13>
0000 1111 1111	256MB	256KB	T_BASE<32:18>:pci_ad<27:13>
0001 1111 1111	512MB	512KB	T_BASE<32:19>:pci_ad<28:13>
0011 1111 1111	1GB	1MB	T_BASE<32:20>:pci_ad<29:13>
0111 1111 1111	2GB <sup>1</sup>	2MB	T_BASE<32:21>:pci_ad<30:13>
1111 1111 1111	4GB <sup>1</sup>	4MB	T_BASE<32:22>:pci_ad<31:13>

<sup>1</sup>This size is not supported by the Alpha PCI 164-266/300.

Figure 4–7 shows the entire translation from PCI address to physical address on a window that implements scatter-gather mapping.

## 4.2 Alpha 21164 Address Mapping to PCI Space

Figure 4-7 PCI to System Bus Scatter-Gather Address Translation Map



MK-2306-11

## 4.2 Alpha 21164 Address Mapping to PCI Space

An 8-entry translation-lookaside buffer (TLB) is provided in the CIA chip for scatter-gather map entries. The TLB is a fully associative cache and holds the eight most recent scatter-gather map lookups. Four of these entries can be “locked,” thus preventing their displacement by the hardware TLB-miss handler. Each of the eight TLB entries holds a PCI address for the tag, and four consecutive 8KB CPU page addresses as the TLB data.

Each time an incoming PCI address hits in a PCI target window that has scatter-gather enabled, bits <30:15> of the PCI address are compared with the 32KB PCI page address in the TLB tag. If a match is found, the required CPU page address is one of the four data entries of the matching TLB entry. PCI address bits <15:13> select the correct 8KB CPU page entry from the four.

If no match is found in the TLB, the scatter-gather map lookup is performed and four page table entries (PTEs) are fetched and written over an existing entry in the TLB. The TLB entry replaced is determined by a round-robin algorithm on the “unlocked” entries. TLB coherency is the responsibility of software by writing to the SG\_TBIA invalidate CSR.

The tag portion of the TLB also contains a DAC flag to indicate that the PCI tag address corresponds to a 64-bit DAC address. Refer to the *DECchip 21171 Core Logic Chipset Technical Reference Manual* for more information.

The process for a scatter-gather TLB hit is as follows:

1. The window compare logic determines if the PCI address has hit in one of the four windows. The PCI\_BASE<SG> bit determines if the scatter-gather path should be taken. If window three has DAC mode enabled, and the PCI cycle is a DAC cycle, then further comparisons are made. Refer to the *DECchip 21171 Core Logic Chipset Technical Reference Manual* for more information on DAC mode.
2. PCI address bits <31:13> are sent to the TLB tag comparator. If the address and DAC bits match in the TLB, then the corresponding CPU 8KB page address is read from the TLB. If this entry is valid, then a TLB hit has occurred and the page address is concatenated with PCI address bits <12:2> to form the physical memory address. If the data entry is invalid, or if the tag compare failed, then a TLB miss has occurred.

The process for a scatter-gather TLB miss is as follows:

1. The relevant bits of the PCI address (as determined by the window mask register) are concatenated with the relevant translated base register bits to form the address used to access the scatter-gather map entry from a table located in system memory.



## 4.2 Alpha 21164 Address Mapping to PCI Space

2. Bits <20:1> of the map entry (PTE) are used to generate the physical page address. This address is appended to the page offset to generate the physical memory address. The TLB is also updated with the four PTE entries that correspond to the 32KB PCI page address, which first missed the TLB. The tag portion of the TLB is loaded with this PCI page address, and the DAC bit is set if the PCI cycle was a DAC cycle.
3. If the requested PTE is marked invalid (bit 0 clear), then a TLB invalid entry exception is taken.

The DECchip 21171 chipset provides support for PC compatibility addressing and holes via PCI bus signal MEMCS#. This allows certain ISA devices to respond to hardwired memory addresses. For more information, refer to the *DECchip 21171 Core Logic Chipset Technical Reference Manual*.



---

## Power and Environmental Requirements

This chapter describes the motherboard's power and environmental requirements, and physical board parameters.

### 5.1 Power Requirements

The Alpha PCI 164-266/300 derives its main dc power from a user-supplied power supply. The board has a total power dissipation of 116 W, excluding any plug-in PCI and ISA devices. Table 5–1 lists the power requirement for each dc supply voltage.

The power supply must supply a **dcok** signal to the system reset logic. Refer to Section 3.8 for additional information.

**Table 5–1 Power Supply dc Current Requirements**

Voltage	Current <sup>1</sup>
+3.3 V dc	16.0 A
+5 V dc	10.0 A
–5 V dc	0 A
+12 V dc	1.0 A
–12 V dc	100.0 mA

<sup>1</sup>Values indicated are for a fully populated Alpha PCI 164-266/300 module excluding plug-in PCI and ISA devices, with a CPU clock speed of 266 MHz.

---

**Caution: Fan Sensor Required**

The 21164 cooling fan *must* have a built-in sensor that will drive a signal if the airflow stops. The sensor is connected to Alpha PCI 164-266/300 board connector J30. When the signal is generated, it resets the system.

---

## 5.2 Environmental Requirements

### 5.2 Environmental Requirements

The 21164 microprocessor is cooled by a small fan blowing directly into the chip's heat sink. The Alpha PCI 164-266/300 motherboard is designed to run efficiently using only this fan. Additional fans may be necessary depending upon cabinetry and I/O board requirements.

The Alpha PCI 164-266/300 is specified to run within the following environment:

Parameter	Specification
Operating temperature	10°C to 40°C (50°F to 104°F)
Storage temperature	-55°C to 125°C (-67°F to 257°F)
Relative humidity	10% to 90% with maximum wet bulb temperature 28°C (82°F) and minimum dew point 2°C (36°F)
Rate of (dry bulb) temperature change	11°C/hour $\pm$ 2°C/hour (20°F/hour $\pm$ 4°F/hour)

### 5.3 Physical Board Parameters

The Alpha PCI 164-266/300 board consists of a 6-layer printed-wiring board (PWB) with components mounted to side 1 only. The board is populated with integrated circuit packages together with supporting active and passive components. The Alpha PCI 164-266/300 is a standard, full-size PC AT board with the following dimensions:

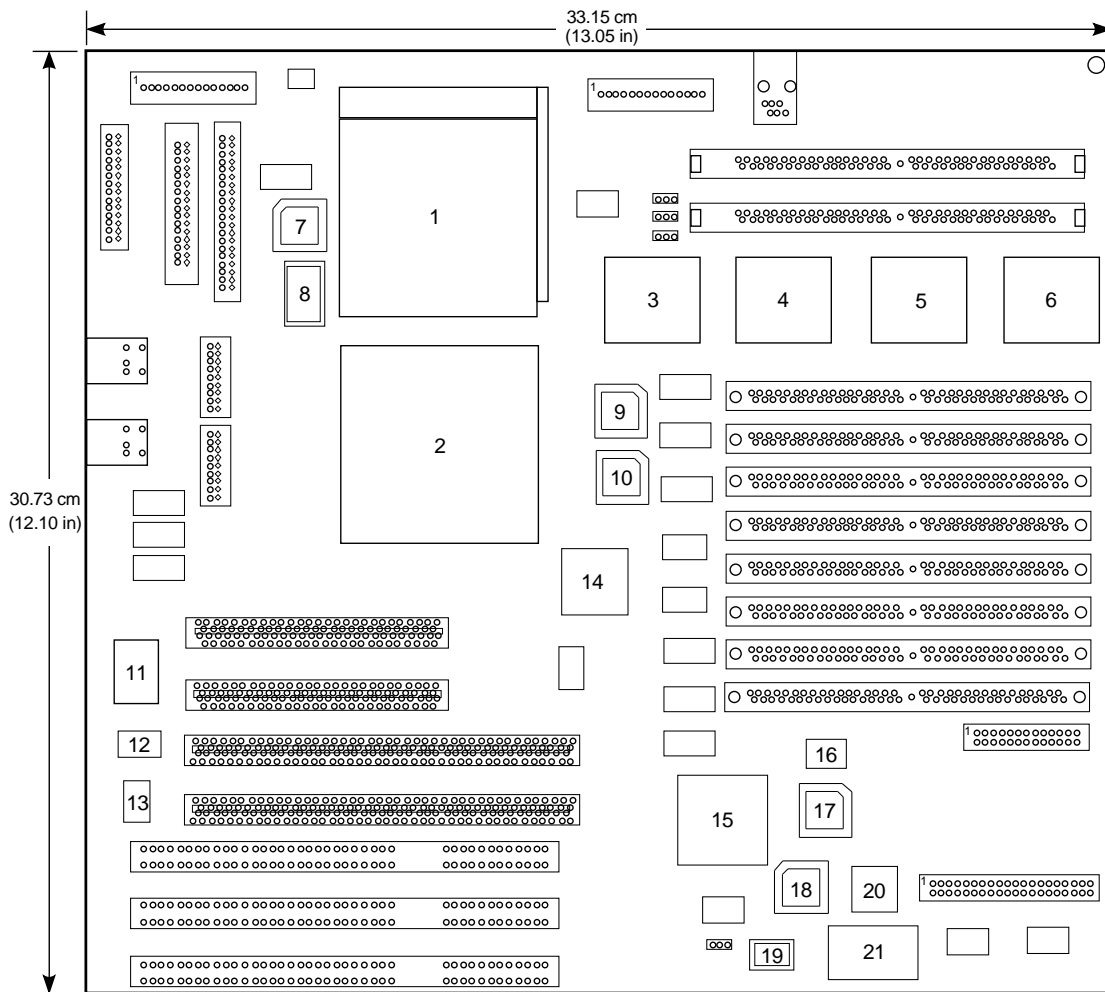
- Width: 30.73 cm (12.10 in  $\pm$ 0.0005 in)
- Length: 33.15 cm (13.05 in  $\pm$ 0.0005 in)
- Height: 6.0 cm (2.375 in)

The board can be used in certain desktop and deskside systems that have adequate clearance for the 21164 heat sink and fan. All ISA and PCI expansion slots are usable in standard desktop or deskside enclosures.

Figure 5-1 shows the board and component outlines, and identifies the major components. Table 5-2 lists the components. Refer to Chapter 2 for jumper and connector functions and locations.

### 5.3 Physical Board Parameters

Figure 5-1 Board Component Layout



MK-2306-32

### 5.3 Physical Board Parameters

**Table 5–2 Board Component List**

<b>Locator Number</b>	<b>Component Number</b>	<b>Component Description</b>
1	U42	Alpha 21164 microprocessor
2	U41	DECchip 21171-CA control, I/O interface, and address (CIA) chip
3	U32	DECchip 21171-BA data switch (DSW0) chip
4	U15	DECchip 21171-BA data switch (DSW2) chip
5	U10	DECchip 21171-BA data switch (DSW1) chip
6	U2	DECchip 21171-BA data switch (DSW3) chip
7	U50	TriQuint TQ2061 phase-locked loop (PLL) clock chip
8	U49	26.66 or 30.00 MHz clock oscillator
9	U31	Main memory row address strobe (RAS) PAL
10	U30	Main memory row address strobe (RAS) PAL
11	U58	National 87312 combination chip
12	U57	ISA clock frequency generator chip
13	X2	14.318-MHz clock oscillator
14	U33	Motorola 88PL117 phase-locked loop (PLL) system clock chip
15	U17	Intel 82378ZB PCI-to-ISA SIO bridge chip
16	U12	Xilinx serial ROM (initialization code) chip
17	U11	MACH210A interrupt request and PCI arbiter PAL
18	U13	Ubus decoder PAL
19	U14	Flash ROM chip
20	U7	Intel 8242 mouse and keyboard controller chip
21	U6	Dallas Semiconductor DS1287 time-of-year (TOY) clock chip

# A

---

## I/O Space Address Maps

This appendix provides lists of the physical Alpha PCI 164-266/300 I/O space assignments, including CIA operating register address space maps and PCI/ISA device register maps. Refer to Chapter 4 for detailed information on sparse/dense space and address translation. The lists include only that portion that is unique to the Alpha PCI 164-266/300 and that affects or reflects the system environment. For full descriptions of all Alpha PCI 164-266/300 registers refer to the *Alpha 21164 Microprocessor Hardware Reference Manual*, the *DECchip 21171 Core Logic Chipset Technical Reference Manual*, and applicable manufacturer's chip data sheets.

### A.1 PCI Sparse Memory Space

There are three regions in the PCI sparse memory contiguous CPU address space:

- Region 0 occupies physical addresses 80.0000.0000 through 83.FFFF.FFFF.
- Region 1 occupies physical addresses 84.0000.0000 through 84.FFFF.FFFF.
- Region 2 occupies physical addresses 85.0000.0000 through 85.7FFF.FFFF.

Refer to Section 4.2.2 for additional information.

### A.2 PCI Sparse I/O Space

There are two regions in the PCI sparse I/O contiguous CPU address space:

- Region A occupies physical addresses 85.8000.0000 through 85.BFFF.FFFF.
- Region B occupies physical addresses 85.C000.0000 through 85.FFFF.FFFF.

Refer to Section 4.2.4 for additional information on PCI sparse I/O space.

## A.2 PCI Sparse I/O Space

### A.2.1 PCI Sparse I/O Space—Region A

PCI sparse I/O space—region A—occupies physical addresses 85.8000.0000 through 85.BFFF.FFFF. The ISA devices are included in this space. Sections A.2.1.1 through A.2.1.4 list the ISA device address maps.

#### A.2.1.1 PC87312 Combination Controller Register Address Space

Table A–1 lists the base address values for the PC87312 combination integrated drive electronics (IDE), diskette, serial port, and parallel port controller.

The general registers are located at addresses 398 (index address) and 399 (data address). For example, writing an index value of 1 to address 398 selects the function address register. If a read operation from address 399 follows, the data associated with the function address register is returned. If a write operation to address 399 follows, the function address register will be updated.

**Table A–1 PC87312 Combination Controller Register Address Space Map**

Address Offset Read/Write	Physical Address	Register
<b>General Registers</b>		
398	85.8000.7300	Index address
399	85.8000.7320	Data address
	<b>Index</b>	<b>Register</b>
	0	Function enable
	1	Function address
	2	Power and test

(continued on next page)



## A.2 PCI Sparse I/O Space

**Table A–1 (Cont.) PC87312 Combination Controller Register Address Space Map**

Address Offset Read/Write	Physical Address	Register
<b>IDE Drive Registers</b>		
1F0-R/W	85.8000.3E00	Data
1F1-R	85.8000.3E20	Error
1F1-W	85.8000.3E20	Features (write precomp)
1F2-R/W	85.8000.3E40	Sector count
1F3-R/W	85.8000.3E60	Sector number
1F4-R/W	85.8000.3E80	Cylinder low
1F5-R/W	85.8000.3EA0	Cylinder high
1F6-R/W	85.8000.3EC0	Drive/head
1F7-R	85.8000.3EE0	Status
1F7-W	85.8000.3EE0	Command
3F6-R	85.8000.7EC0	Alternate status
3F6-W	85.8000.7EC0	Device control
3F7-R	85.8000.7EE0	Drive address
3F7-W	85.8000.7EE0	None (tristate bus)

(continued on next page)

## A.2 PCI Sparse I/O Space

**Table A–1 (Cont.) PC87312 Combination Controller Register Address Space Map**

Address Offset Read/Write	Physical Address	Register
<b>COM2 Serial Port Registers</b>		
2F8-R 0DLAB=0	85.8000.5F00	COM2 receiver buffer
2F8-W 0DLAB=0	85.8000.5F00	COM2 transmitter holding
2F8 0DLAB=1	85.8000.5F00	COM2 divisor latch (LSB)
2F9 1DLAB=0	85.8000.5F20	COM2 interrupt enable
2F9 1DLAB=1	85.8000.5F20	COM2 divisor latch (MSB)
2FA-R	85.8000.5F40	COM2 interrupt identification
2FA-W	85.8000.5F40	COM2 FIFO control
2FB	85.8000.5F60	COM2 line control
2FC	85.8000.5F80	COM2 modem control
2FD	85.8000.5FA0	COM2 line status
2FE	85.8000.5FC0	COM2 modem status
2FF	85.8000.5FE0	COM2 scratch pad
<b>Parallel Port Registers</b>		
3BC-R/W	85.8000.7780	Data
3BD-R	85.8000.77A0	Status
3BE-R/W	85.8000.77C0	Control
3BF	85.8000.77E0	None (tristate)

(continued on next page)

## A.2 PCI Sparse I/O Space

**Table A–1 (Cont.) PC87312 Combination Controller Register Address Space Map**

<b>Address Offset Read/Write</b>	<b>Physical Address</b>	<b>Register</b>
<b>Diskette Registers</b>		
3F0-R	85.8000.7E00	Status A
3F1-R	85.8000.7E20	Status B
3F2-R/W	85.8000.7E40	Digital output
3F3-R/W	85.8000.7E60	Tape drive
3F4-R	85.8000.7E80	Main status
3F4-W	85.8000.7E80	Data rate select
3F5-R/W	85.8000.7EA0	Data (FIFO)
3F6	85.8000.7EC0	None (tristate bus)
3F7-R	85.8000.7EE0	Digital input
3F7-W	85.8000.7EE0	Configuration control
<b>COM1 Serial Port Registers</b>		
3F8-R 0DLAB=0	85.8000.7F00	COM1 receiver buffer
3F8-W 0DLAB=0	85.8000.7F00	COM1 transmitter holding
3F8 0DLAB=1	85.8000.7F00	COM1 divisor latch (LSB)
3F9 1DLAB=0	85.8000.7F20	COM1 interrupt enable
3F9 1DLAB=1	85.8000.7F20	COM1 divisor latch (MSB)
3FA-R	85.8000.7F40	COM1 interrupt identification
3FA-W	85.8000.7F40	COM1 FIFO control
3FB	85.8000.7F60	COM1 line control
3FC	85.8000.7F80	COM1 modem control
3FD	85.8000.7FA0	COM1 line status
3FE	85.8000.7FC0	COM1 modem status
3FF	85.8000.7FE0	COM1 scratch pad

## A.2 PCI Sparse I/O Space

### A.2.1.2 8242AH Keyboard and Mouse Controller Addresses

Table A–2 lists the register and memory addresses for the keyboard/mouse controller.

**Table A–2 Keyboard and Mouse Controller Addresses**

Offset	Physical Address	Register
60-R	85.8000.0C00	Auxiliary/keyboard data
60-W	85.8000.0C00	Command data
64-R	85.8000.0C80	Read status
64-W	85.8000.0C80	Command

### A.2.1.3 Time-of-Year (TOY) Clock Addresses

Table A–3 lists the register and memory addresses for the TOY clock device. The time-of-year clock register is accessed by writing to address 70 with the latched index. Then, reading from, or writing to, address 71 reads or writes the register. For example, writing an 8 to address 70 followed by a read operation from address 71 returns the value of the month. Writing a 4 to address 70 followed by a write operation to address 71 updates the hour register.

## A.2 PCI Sparse I/O Space

**Table A–3 Time-of-Year Clock Device Addresses**

Offset	Latched Index	Physical Address	Register
70	0	85.8000.0E00	Seconds
70	1	85.8000.0E00	Seconds alarm
70	2	85.8000.0E00	Minutes
70	3	85.8000.0E00	Minutes alarm
70	4	85.8000.0E00	Hour
70	5	85.8000.0E00	Hour alarm
70	6	85.8000.0E00	Day of week
70	7	85.8000.0E00	Day of month
70	8	85.8000.0E00	Month
70	9	85.8000.0E00	Year
70	A	85.8000.0E00	Register A
70	B	85.8000.0E00	Register B
70	C	85.8000.0E00	Register C
70	D	85.8000.0E00	Register D
71	—	85.8000.0E20	TOY clock chip select

### A.2.1.4 Flash ROM Segment Select Register

The flash ROM is partitioned into two 512KB segments. The segments are selected by **flash\_adr19**. To select the first 512KB segment, write a value of 0 to ISA port address 0x800<sub>16</sub>. To access the second 512KB segment, write a value of 1 to this register. Alternatively, using the debug monitor, you can type command `wb 800 0` or `wb 800 1`.

Table A–4 lists the register address for the flash ROM segment select register. This register is write-only. Refer to Section A.3.1 for dense space flash ROM memory addresses.

**Table A–4 Flash ROM Segment Select Register**

Offset	Physical Address	Register
x800	85.8001.0000	Flash ROM segment select

## A.2 PCI Sparse I/O Space

### A.2.1.5 Configuration Jumpers (CONF4—CONF15)

Reading the addresses listed in Table A-5 returns the value of the configuration jumpers CONF4 through CONF15. Bits corresponding to CONF0 through CONF3 are hardwired to the presence detect signals from the DRAM SIMMs.

**Table A-5 Configuration Jumpers (CONF4—CONF15)**

Offset	Physical Address	Description
x801	85.8001.0020	Bits <3:0> are presence detect signals <PD4:PD1>. Bits <7:4> are CONF<7:4>.
x802	85.8001.0040	Bits <7:0> are CONF<15:8>.

### A.2.1.6 Interrupt Control PLD Addresses

Table A-6 lists the registers and memory addresses for the interrupt control programmable logic device (PLD).

**Table A-6 Interrupt Control PLD Addresses**

Offset	Physical Address	Register
x804	85.8001.0080	Interrupt status/interrupt mask 1
x805	85.8001.00A0	Interrupt status/interrupt mask 2
x806	85.8001.00C0	Interrupt status/interrupt mask 3

## A.2.2 PCI Sparse I/O Space—Region B

PCI sparse I/O space—region B—occupies physical addresses 85.C000.0000 through 85.FFFF.FFFF. This region includes the PCI-to-ISA bridge operating register address space as well as the operating registers for any optional PCI plug-in boards. Table A-7 is a map of the SIO PCI-to-ISA bridge operating address space.

## A.2 PCI Sparse I/O Space

**Table A-7 SIO PCI-to-ISA Bridge Operating Register Address Space Map**

Offset	Address	Register
000	85.C000.0000	DMA1 CH0 Base and Current Address
001	85.C000.0020	DMA1 CH0 Base and Current Count
002	85.C000.0040	DMA1 CH1 Base and Current Address
003	85.C000.0060	DMA1 CH1 Base and Current Count
004	85.C000.0080	DMA1 CH2 Base and Current Address
005	85.C000.00A0	DMA1 CH2 Base and Current Count
006	85.C000.00C0	DMA1 CH3 Base and Current Address
007	85.C000.00E0	DMA1 CH3 Base and Current Count
008	85.C000.0100	DMA1 Status and Command
009	85.C000.0120	DMA1 Write Request
00A	85.C000.0140	DMA1 Write Single Mask Bit
00B	85.C000.0160	DMA1 Write Mode
00C	85.C000.0180	DMA1 Clear Byte Pointer
00D	85.C000.01A0	DMA1 Master Clear
00E	85.C000.01C0	DMA1 Clear Mask
00F	85.C000.01E0	DMA1 Read/Write All Mask Register Bits
020	85.C000.0400	INT 1 Control
021	85.C000.0420	INT 1 Mask
040	85.C000.0800	Timer Counter 1 - Counter 0 Count
041	85.C000.0820	Timer Counter 1 - Counter 1 Count
042	85.C000.0840	Timer Counter 1 - Counter 2 Count
043	85.C000.0860	Timer Counter 1 - Command Mode
060	85.C000.0C00	Reset UBus IRQ12
061	85.C000.0C20	NMI Status and Control
070	85.C000.0E00	CMOS RAM Address and NMI Mask
078-07B	85.C000.0F18	BIOS Timer
080	85.C000.1000	DMA Page Register Reserved
081	85.C000.1020	DMA Channel 2 Page
082	85.C000.1040	DMA Channel 3 Page

(continued on next page)

## A.2 PCI Sparse I/O Space

**Table A-7 (Cont.) SIO PCI-to-ISA Bridge Operating Register Address Space Map**

Offset	Address	Register
083	85.C000.1060	DMA Channel 1 Page
084	85.C000.1080	DMA Page Register Reserved
085	85.C000.10A0	DMA Page Register Reserved
086	85.C000.10C0	DMA Page Register Reserved
087	85.C000.10E0	DMA Channel 0 Page
088	85.C000.1100	DMA Page Register Reserved
089	85.C000.1120	DMA Channel 6 Page
08A	85.C000.1140	DMA Channel 7 Page
08B	85.C000.1160	DMA Channel 5 Page
08C	85.C000.1180	DMA Page Register Reserved
08D	85.C000.11A0	DMA Page Register Reserved
08E	85.C000.11C0	DMA Page Register Reserved
08F	85.C000.11E0	DMA Low Page Register Refresh
090	85.C000.1200	DMA Page Register Reserved
092	85.C000.1240	Port 92
094	85.C000.1280	DMA Page Register Reserved
095	85.C000.12A0	DMA Page Register Reserved
096	85.C000.12C0	DMA Page Register Reserved
098	85.C000.1300	DMA Page Register Reserved
09C	85.C000.1380	DMA Page Register Reserved
09D	85.C000.13A0	DMA Page Register Reserved
09E	85.C000.13C0	DMA Page Register Reserved
09F	85.C000.13E0	DMA Low Page Register Refresh
0A0	85.C000.1400	INT2 Control
0A1	85.C000.1420	INT2 Mask
0C0	85.C000.1800	DMA2 CH0 Base and Current Address
0C2	85.C000.1840	DMA2 CH0 Base and Current Count
0C4	85.C000.1880	DMA2 CH1 Base and Current Address

(continued on next page)



## A.2 PCI Sparse I/O Space

**Table A-7 (Cont.) SIO PCI-to-ISA Bridge Operating Register Address Space Map**

Offset	Address	Register
0C6	85.C000.18C0	DMA2 CH1 Base and Current Count
0C8	85.C000.1900	DMA2 CH2 Base and Current Address
0CA	85.C000.1940	DMA2 CH2 Base and Current Count
0CC	85.C000.1980	DMA2 CH3 Base and Current Address
0CE	85.C000.19C0	DMA2 CH3 Base and Current Count
0D0	85.C000.1A00	DMA2 Status(r) and Command(w)
0D2	85.C000.1A40	DMA2 Write Request
0D4	85.C000.1A80	DMA2 Write Single Mask Bit
0D6	85.C000.1AC0	DMA2 Write Mode
0D8	85.C000.1B00	DMA2 Clear Byte Pointer
0DA	85.C000.1B40	DMA2 Master Clear
0DC	85.C000.1B80	DMA2 Clear Mask
0DE	85.C000.1BC0	DMA2 Read/Write All Mask Register Bits
0F0	85.C000.1E00	Coprocessor Error
372	85.C000.6E40	Secondary Floppy Disk Digital Output
3F2	85.C000.7E40	Primary Floppy Disk Digital Output
40A	85.C000.8140	Scatter/Gather Interrupt Status
40B	85.C000.8160	DMA1 Extended Mode
410	85.C000.8200	CH0 Scatter/Gather Command
411	85.C000.8220	CH1 Scatter/Gather Command
412	85.C000.8240	CH2 Scatter/Gather Command
413	85.C000.8260	CH3 Scatter/Gather Command
415	85.C000.82A0	CH5 Scatter/Gather Command
416	85.C000.82C0	CH6 Scatter/Gather Command
417	85.C000.82E0	CH7 Scatter/Gather Command
418	85.C000.8300	CH0 Scatter/Gather Status
419	85.C000.8320	CH1 Scatter/Gather Status
41A	85.C000.8340	CH2 Scatter/Gather Status

(continued on next page)

## A.2 PCI Sparse I/O Space

**Table A-7 (Cont.) SIO PCI-to-ISA Bridge Operating Register Address Space Map**

Offset	Address	Register
41B	85.C000.8360	CH3 Scatter/Gather Status
41D	85.C000.83A0	CH5 Scatter/Gather Status
41E	85.C000.83C0	CH6 Scatter/Gather Status
41F	85.C000.83E0	CH7 Scatter/Gather Status
420-423	85.C000.8418	CH0 Scatter/Gather Descriptor Table Pointer
424-427	85.C000.8498	CH1 Scatter/Gather Descriptor Table Pointer
428-42B	85.C000.8518	CH2 Scatter/Gather Descriptor Table Pointer
42C-42F	85.C000.8598	CH3 Scatter/Gather Descriptor Table Pointer
434-437	85.C000.8698	CH5 Scatter/Gather Descriptor Table Pointer
438-43B	85.C000.8718	CH6 Scatter/Gather Descriptor Table Pointer
43C-43F	85.C000.8798	CH7 Scatter/Gather Descriptor Table Pointer
481	85.C000.9020	DMA CH2 High Page
482	85.C000.9040	DMA CH3 High Page
483	85.C000.9060	DMA CH1 High Page
487	85.C000.90E0	DMA CH0 High Page
489	85.C000.9120	DMA CH6 High Page
48A	85.C000.9140	DMA CH7 High Page
48B	85.C000.9160	DMA CH5 High Page
4D6	85.C000.9AC0	DMA2 Extended Mode

## A.3 PCI Dense Memory Space

### A.3 PCI Dense Memory Space

PCI dense memory space occupies physical addresses 86.0000.0000 through 86.FFFF.FFFF and is typically used for PCI data buffers (such as a video frame buffer). Refer to Section 4.2.5 for additional information.

#### A.3.1 Flash ROM Memory Addresses

Table A–8 lists the address range for the flash ROM. Refer to Section A.2.1.4 for details on selecting one of two flash ROM segments.

**Table A–8 Flash ROM Memory Addresses (Within Segment)**

Offset	Physical Address	Capacity
0.0000–7.FFFF	86.FFF8.0000–86.FFFF.FFFF	512KB

#### A.3.2 Map of Flash ROM Memory

Table A–9 provides a map of flash ROM memory.

**Table A–9 Map of Flash ROM Memory**

Offset	Physical Address <sup>1</sup>	Block Number <sup>2</sup>	Capacity
0.0000–0.FFFF	86.FFF8.0000–86.FFF8.FFFF	0,8	64KB
1.0000–1.FFFF	86.FFF9.0000–86.FFF9.FFFF	1,9	64KB
2.0000–2.FFFF	86.FFFA.0000–86.FFFA.FFFF	2,10	64KB
3.0000–3.FFFF	86.FFFB.0000–86.FFFB.FFFF	3,11	64KB
4.0000–4.FFFF	86.FFFC.0000–86.FFFC.FFFF	4,12	64KB
5.0000–5.FFFF	86.FFFD.0000–86.FFFD.FFFF	5,13	64KB
6.0000–6.FFFF	86.FFFE.0000–86.FFFE.FFFF	6,14	64KB
7.0000–7.FFFF	86.FFFF.0000–86.FFFF.FFFF	7,15	64KB

<sup>1</sup>Dense space addresses. Byte accesses are not possible using this space. Use sparse space for finer granularity.

<sup>2</sup>The block number is determined by the value in the flash ROM segment select register (see Section A.2.1.4).

## A.3 PCI Dense Memory Space

### A.3.3 Flash ROM Configuration Registers

Table A–10 lists the configuration registers for the Intel 28F008SA 1MB flash ROM. A read operation is performed by reading from the appropriate address.

To write data, the flash ROM must first be erased. The structure of the flash ROM allows only the flash ROM to be erased in 64KB blocks (see Section A.3.2).

In order to change one byte, the following steps must be completed:

1. Read the entire 64KB block into system memory.
2. Change the desired byte in system memory.
3. Erase the 64KB block in flash ROM.
4. Write the entire 64KB block from system memory to the flash ROM.

---

**Note**

---

In order to write to flash ROM, Jumper J14 (protect/enable) must be positioned on pins 2 and 3 (see Table 2–1).

---

All flash ROM accesses (except for read operations) require two bus cycles. During the first cycle, register data is written to set up the registers. During the second cycle, the read or write transaction performs the operation desired. For more information about reading, erasing, and writing the flash ROM, see the Intel *Flash Memory* document.

Accessing the flash ROM registers requires byte access, which is only possible through use of PCI sparse memory space. The Alpha PCI 164-266/300 flash ROM resides in PCI memory address range FFF8.0000 to FFFF.FFFF. See Section 4.2.2 for information about accessing this address range through sparse memory space.

### A.3 PCI Dense Memory Space

**Table A–10 Flash ROM Configuration Registers**

Offset	Data Written on First Access	Register
X <sup>1</sup>	FF	Read array/reset register
X	90	Intelligent identifier register
X	70	Read status register
X	50	Clear status register
BA <sup>2</sup>	20	Erase setup/confirm register
X	B0	Erase suspend/resume register
WA <sup>3</sup>	40	Byte write setup/write register
WA	10	Alternate byte write setup/write register

<sup>1</sup>X = Any byte within the flash ROM address range.

<sup>2</sup>BA = Target address within the block being erased.

<sup>3</sup>WA = Target address of write transaction to memory.

## A.4 PCI Configuration Address Space

### A.4 PCI Configuration Address Space

The PCI configuration address space occupies physical addresses 87.0000.0000 through 87.1FFF.FFFF. The PCI configuration register set occupies this space. A read or write access to this space causes a configuration read or write cycle on the PCI. Table A-11 identifies the Alpha PCI 164-266/300 PCI devices and the corresponding PCI address bit that drives the device's **IDSEL** pin. Refer to Section 4.2.6 for additional information on this space.

**Table A-11 Address Bits and PCI Device IDSEL Pins**

PCI Device	PCI Address Bit Driving IDSEL Pin	Physical Address
PCI expansion slot 2 (J24)	<b>pci_ad&lt;16&gt;</b>	87.0005.0000
PCI expansion slot 0 (J22)	<b>pci_ad&lt;17&gt;</b>	87.0006.0000
PCI expansion slot 1 (J23)	<b>pci_ad&lt;18&gt;</b>	87.0007.0000
SIO bridge	<b>pci_ad&lt;19&gt;</b>	87.0008.0000
PCI expansion slot 3 (J25)	<b>pci_ad&lt;20&gt;</b>	87.0009.0000

#### A.4.1 SIO PCI-to-ISA Bridge Configuration Address Space

Table A-12 is a map of SIO PCI-to-ISA bridge configuration address space. PCI address bit **pci\_ad19** drives the **idsel** chip select pin for access to the configuration register space.

## A.4 PCI Configuration Address Space

**Table A–12 SIO PCI-to-ISA Bridge Configuration Address Space Map**

Offset	Address	Register
00–01	87.0008.0008	Vendor ID
02–03	87.0008.0048	Device ID
04–05	87.0008.0088	Command
06–07	87.0008.00C8	Device Status
08	87.0008.0100	Revision ID
40	87.0008.0800	PCI Control
41	87.0008.0820	PCI Arbiter Control
42	87.0008.0840	PCI Arbiter Priority Control
44	87.0008.0880	MEMCS# Control
45	87.0008.08A0	MEMCS# Bottom of Hole
46	87.0008.08C0	MEMCS# Top of Hole
47	87.0008.08E0	MEMCS# Top of Memory
48	87.0008.0900	ISA Address Decoder Control
49	87.0008.0920	ISA Address Decoder ROM Block Enable
4A	87.0008.0940	ISA Address Decoder Bottom of Hole
4B	87.0008.0960	ISA Address Decoder Top of Hole
4C	87.0008.0980	ISA Controller Recovery Timer
4D	87.0008.09A0	ISA Clock Divisor
4E	87.0008.09C0	Utility Bus Chip Select Enable A
4F	87.0008.09E0	Utility Bus Chip Select Enable B
54	87.0008.0A80	MEMCS# Attribute Register #1
55	87.0008.0AA0	MEMCS# Attribute Register #2
56	87.0008.0AC0	MEMCS# Attribute Register #3
57	87.0008.0AE0	Scatter/Gather Relocation Base Address
80–81	87.0008.1008	BIOS Timer Base Address

## A.5 PCI Interrupt Acknowledge/Special Cycle Address Space

## A.5 PCI Interrupt Acknowledge/Special Cycle Address Space

This space occupies physical addresses 87.2000.0000 through 87.3FFF.FFFF. Refer to Section 4.2.7 for additional information.

## A.6 Hardware-Specific and Miscellaneous Register Space

This space occupies physical addresses 87.4000.0000 through 87.6FFF.FFFF and covers the DECchip 21171-CA (CIA) address space. Registers accessed in this space use a hardware-specific variant of sparse space encoding. CPU address bits <27:6> are used as a longword address. CPU address bits <5:0> must be zero. All CIA registers are accessed with longword granularity.

### A.6.1 CIA Main CSR Space

This space occupies physical addresses 87.4000.0000 through 87.4FFF.FFFF. Table A–13 lists all the CIA chip's general control, diagnostic, and error registers.

**Table A–13 CIA Control, Diagnostic, and Error Registers**

Register	Type	Address	Description
PCI_REV	RO	87.4000.0080	PCI revision (CIA revision)
PCI_LAT	RO	87.4000.00C0	PCI latency
CIA_CTRL	RW	87.4000.0100	CIA control register
CIA_CNFG	RO	87.4000.0200	CIA configuration register
HAE_MEM	RW	87.4000.0400	Hardware address extension for sparse memory
HAE_IO	RW	87.4000.0440	Hardware address extension for sparse I/O
CFG	RW	87.4000.0480	PCI configuration register
CACK_EN	RW	87.4000.0600	CIA acknowledgement control register
CIA_DIAG	RW	87.4000.2000	CIA diagnostic control register
DIAG_CHECK	RW	87.4000.3000	Diagnostic check register
PERF_MONITOR	RO	87.4000.4000	Performance monitor register
PERF_CONTROL	RW	87.4000.4040	Performance control register
CPU_ERR0	RO	87.4000.8000	CPU error information register 0

(continued on next page)



## A.6 Hardware-Specific and Miscellaneous Register Space

**Table A–13 (Cont.) CIA Control, Diagnostic, and Error Registers**

Register	Type	Address	Description
CPU_ERR1	RO	87.4000.8040	CPU error information register 1
CIA_ERR	R/WC	87.4000.8200	CIA error register
CIA_STAT	RW	87.4000.8240	CIA status register
ERR_MASK	R/WC	87.4000.8280	CIA error mask register
CIA_SYN	RO	87.4000.8300	CIA syndrome register
CIA_MEM0	RO	87.4000.8400	CIA memory port status register 0
CIA_MEM1	RO	87.4000.8440	CIA memory port status register 1
PCI_ERR0	R/WC	87.4000.8800	PCI error status register 0
PCI_ERR1	R/WC	87.4000.8840	PCI error status register 1

### A.6.2 CIA Memory Control CSR Space

CIA memory control CSR space occupies physical addresses 87.5000.0000 through 87.5FFF.FFFF. Table A–14 lists all the CIA chip's memory control registers.

**Table A–14 CIA Memory Control Registers**

Register	Type	Address	Description
MCR	RW	87.5000.0000	Memory configuration register
MBA0	RW	87.5000.0600	Memory base address register 0
MBA2	RW	87.5000.0680	Memory base address register 2
MBA4	RW	87.5000.0700	Memory base address register 4
MBA6	RW	87.5000.0780	Memory base address register 6
MBA8	RW	87.5000.0800	Memory base address register 8
MBAA	RW	87.5000.0880	Memory base address register 10
MBAC	RW	87.5000.0900	Memory base address register 12
MBAE	RW	87.5000.0980	Memory base address register 14
TMG0	RW	87.5000.0B00	Memory timing information base register 0
TMG1	RW	87.5000.0B40	Memory timing information base register 1
TMG2	RW	87.5000.0B80	Memory timing information base register 2

## A.6 Hardware-Specific and Miscellaneous Register Space

### A.6.3 CIA PCI Address Translation Map Space

CIA PCI address translation map space occupies physical addresses 87.6000.0000 through 87.6FFF.FFFF. Table A–15 lists all the CIA chip's PCI address translation registers.

**Table A–15 PCI Address Translation Registers**

Register	Type	Address	Description
TBIA	WO	87.6000.0100	Scatter-gather translation buffer invalidate register
W0_BASE	RW	87.6000.0400	Window base 0 register
W0_MASK	RW	87.6000.0440	Window mask 0 register
T0_BASE	RW	87.6000.0480	Translated base 0 register
W1_BASE	RW	87.6000.0500	Window base 1 register
W1_MASK	RW	87.6000.0540	Window mask 1 register
T1BASE	RW	87.6000.0580	Translated base 1 register
W2_BASE	RW	87.6000.0600	Window base 2 register
W2_MASK	RW	87.6000.0640	Window mask 2 register
T2_BASE	RW	87.6000.0680	Translated base 2 register
W3_BASE	RW	87.6000.0700	Window base 3 register
W3_MASK	RW	87.6000.0740	Window mask 3 register
T3_BASE	RW	87.6000.0780	Translated base 3 register
W_DAC	RW	87.6000.07C0	Window DAC register
LTB_TAG0	RW	87.6000.0800	Lockable translation buffer tag0
LTB_TAG1	RW	87.6000.0840	Lockable translation buffer tag1
LTB_TAG2	RW	87.6000.0880	Lockable translation buffer tag2
LTB_TAG3	RW	87.6000.08C0	Lockable translation buffer tag3
TB_TAG0	RW	87.6000.0900	Translation buffer tag0
TB_TAG1	RW	87.6000.0940	Translation buffer tag1
TB_TAG2	RW	87.6000.0980	Translation buffer tag2
TB_TAG3	RW	87.6000.09C0	Translation buffer tag3
TB0_PAGE0	RW	87.6000.1000	Translation buffer 0 page0
TB0_PAGE1	RW	87.6000.1040	Translation buffer 0 page1

(continued on next page)

## A.6 Hardware-Specific and Miscellaneous Register Space

**Table A–15 (Cont.) PCI Address Translation Registers**

Register	Type	Address	Description
TB0_PAGE2	RW	87.6000.1080	Translation buffer 0 page2
TB0_PAGE3	RW	87.6000.10C0	Translation buffer 0 page3
TB1_PAGE0	RW	87.6000.1100	Translation buffer 1 page0
TB1_PAGE1	RW	87.6000.1140	Translation buffer 1 page1
TB1_PAGE2	RW	87.6000.1180	Translation buffer 1 page2
TB1_PAGE3	RW	87.6000.11C0	Translation buffer 1 page3
TB2_PAGE0	RW	87.6000.1200	Translation buffer 2 page0
TB2_PAGE1	RW	87.6000.1240	Translation buffer 2 page1
TB2_PAGE2	RW	87.6000.1280	Translation buffer 2 page2
TB2_PAGE3	RW	87.6000.12C0	Translation buffer 2 page3
TB3_PAGE0	RW	87.6000.1300	Translation buffer 3 page0
TB3_PAGE1	RW	87.6000.1340	Translation buffer 3 page1
TB3_PAGE2	RW	87.6000.1380	Translation buffer 3 page2
TB3_PAGE3	RW	87.6000.13C0	Translation buffer 3 page3
TB4_PAGE0	RW	87.6000.1400	Translation buffer 4 page0
TB4_PAGE1	RW	87.6000.1440	Translation buffer 4 page1
TB4_PAGE2	RW	87.6000.1480	Translation buffer 4 page2
TB4_PAGE3	RW	87.6000.14C0	Translation buffer 4 page3
TB5_PAGE0	RW	87.6000.1500	Translation buffer 5 page0
TB5_PAGE1	RW	87.6000.1540	Translation buffer 5 page1
TB5_PAGE2	RW	87.6000.1580	Translation buffer 5 page2
TB5_PAGE3	RW	87.6000.15C0	Translation buffer 5 page3
TB6_PAGE0	RW	87.6000.1600	Translation buffer 6 page0
TB6_PAGE1	RW	87.6000.1640	Translation buffer 6 page1
TB6_PAGE2	RW	87.6000.1680	Translation buffer 6 page2
TB6_PAGE3	RW	87.6000.16C0	Translation buffer 6 page3
TB7_PAGE0	RW	87.6000.1700	Translation buffer 7 page0
TB7_PAGE1	RW	87.6000.1740	Translation buffer 7 page1
TB7_PAGE2	RW	87.6000.1780	Translation buffer 7 page2

(continued on next page)

## A.6 Hardware-Specific and Miscellaneous Register Space

**Table A–15 (Cont.) PCI Address Translation Registers**

Register	Type	Address	Description
TB7_PAGE3	RW	87.6000.17C0	Translation buffer 7 page3

## A.7 Alpha 21164 Microprocessor Cbox IPR Space

The 21164 microprocessor cache control and bus interface unit (Cbox) IPR space occupies physical addresses FF.FFF0.0000 through FF.FFFF.FFFF.

Table A–16 lists three key 21164 registers that configure the internal L2 secondary cache (Scache) and external L3 backup cache (Bcache). For additional information, refer to the *Alpha 21164 Microprocessor Hardware Reference Manual*.

**Table A–16 Alpha 21164 Cache Configuration Registers**

Register	Type	Address	Description
SC_CTL	RW	FF.FFF0.00A8	Scache control register
BC_CONTROL	W	FF.FFF0.0128	Bcache, system interface, and Bcache test control register
BC_CONFIG	W	FF.FFF0.01C8	Bcache configuration register (size and timing)

# B

---

## SROM Initialization

The Alpha 21164 microprocessor provides a mechanism for loading the initial instruction stream (Istream) from a compact serial ROM (SROM) to start the bootstrap procedure. The SROM executable image is limited to the size of the CPU instruction cache (Icache). Because the image is running only in the Icache, it is relatively difficult to debug. Therefore, the scope and purpose of this code is limited to performing the system initialization necessary to boot the next level of firmware contained in the larger system (flash) ROM.

Trade-offs between simplicity and convenience were made to support the Alpha PCI 164-266/300 in various configurations.

### B.1 SROM Initialization

After reset, the contents of the SROM are loaded into the Icache. After loading the Icache, the CPU begins execution at location zero. Execution is performed in the CPU PALmode environment with privileged access to the computer hardware. The general steps performed by the SROM initialization are:

1. Initialize the CPU's internal processor registers (IPRs).
2. Set up internal L1/L2 caches.
3. Perform the minimum I/O subsystem initialization necessary to access the real-time clock (RTC) and the system's flash ROM.
4. Detect CPU speed by polling the periodic interrupt flag (PIF) in the RTC.
5. Set up memory and backup cache (Bcache) parameters based on the speed of the CPU.
6. Wake up the DRAMs.
7. Initialize the Bcache.
8. Copy the contents of the entire system memory to itself to ensure good memory data parity.

## B.1 SROM Initialization

9. Scan the system flash ROM for a special header that specifies where and how the system flash ROM firmware should be loaded.
10. Copy the contents of the system flash ROM to memory and begin code execution.
11. Pass parameters up to the next level of firmware to provide a predictable firmware interface.

## B.2 Firmware Interface

A firmware interface provides a mechanism for passing critical information about the state of the system and CPU up to the next level of firmware. This interface is achieved through the use of a set of defined SROM output parameters as described in Table B-1

This specific firmware interface serves the 21164 microprocessor. Other Alpha microprocessor implementations require a different firmware interface.

**Table B-1 Output Parameter Descriptions**

Output Parameter	Parameter Description
r1 (t0)—BC_CONTROL value	The BC_CONTROL value allows the next-level software to preserve any system-specific Bcache configuration information.
r2 (t1)—BC_CONFIG value	The BC_CONFIG value preserves the Bcache configuration information such as, size and read/write speed.
r3 (t2)—BC_CONFIG_OFF value	The BC_CONFIG value for turning the Bcache off, if necessary. This value may be harder to be determined by the next level of firmware, so the SROM computes it and passes it up.
r17 (a1)—Memory size	This value is an unsigned quadword count of the number of contiguous bytes of good memory in the system starting at physical address zero. This simple mechanism is sufficient for simple systems. Systems that need to communicate more detailed memory configuration can do so through the system context value (see last entry in this table).

(continued on next page)

## B.2 Firmware Interface

**Table B–1 (Cont.) Output Parameter Descriptions**

Output Parameter	Parameter Description
r18 (a2)—Cycle count in picoseconds	This value is the number of picoseconds that elapse for each increment of the processor cycle count (as read by the RPCC instruction). This may be a multiple of the actual internal cycle count of the microprocessor as specified in the <i>Alpha Architecture Reference Manual</i> (a microprocessor will increment the processor cycle count a multiple of the microprocessor clock, where the multiple is a power of 2, including $2^0 = 1$ ).
r19 (a3)—Signature and system revision ID	<p>This register includes a signature that specifies that the transfer is following the standard protocol and that the other values can be trusted. In addition, the signature can identify which version of the protocol is being followed. The system revision is a 16-bit field that communicates system revisions that would be significant to operating system software. The register has the following format:</p> <p style="margin-left: 40px;">Bits &lt;63:32&gt; = Ignore            Bits &lt;31:16&gt; = Signature            Bits &lt;15:0&gt; = System Revision</p> <p>Valid signatures have the following values:</p> <p style="margin-left: 40px;">0xdeca—V1 (previous version of this specification)            0xdecb—V2 (current version of this specification)</p>
r20 (a4)—Active processor mask	<p>The processor mask identifies each processor that is present on the current system. Each mask bit corresponds to a processor number associated by the bit number (for example, bit 0 corresponds to processor 0). A value of 1 in the mask indicates that the processor is present, a value of 0 indicates that the processor is not present.</p> <p>To qualify as present a processor must be:</p> <ul style="list-style-type: none"> <li>• Physically present</li> <li>• Functioning normally</li> <li>• Capable of sending and receiving interprocessor interrupt requests</li> </ul> <p>Uniprocessor systems pass a value of 1 to this register.</p>

(continued on next page)

## B.2 Firmware Interface

**Table B-1 (Cont.) Output Parameter Descriptions**

Output Parameter	Parameter Description
r21 (a5)—System context value	The context value is interpreted in a system-specific manner. If the system needs to pass more than one system-specific parameter, then it may pass a context value. A context value is a physical address pointer to a data structure of many system-specific values.

## B.3 Automatic CPU Speed Detection

The Alpha PCI 164-266/300 real-time clock (RTC) detects the speed of the CPU. This allows a somewhat generic SROM to support Alpha PCI 164-266/300 boards configured for different CPU speeds. The speed is determined by counting CPU cycles between RTC interrupts that are set to occur at known time intervals (1/8 second).

## B.4 CPU Bus Interface Timing

The Alpha PCI 164-266/300 Bcache timing is based on CPU speed in addition to fixed delays associated with the Bcache subsystem. The pertinent Bcache delays used in the calculations result from the logic devices used in the Bcache subsystem, SRAM specifications, and board etch delays. This data is used to calculate the appropriate BIU\_CTL register setting, which determines the CPU pin bus timing.

Table B-2, Table B-3, and Table B-4 describe the fixed delays for the Alpha PCI 164-266/300.



## B.4 CPU Bus Interface Timing

**Table B–2 Cache Loop Delay Characteristics**

Function	Description
Tadr	Index line delay from CPU to SRAM pins
Tbuf	Buffer gate delay
Tdat	Data path delay between CPU and SRAM pins (and vice versa)
Toe1	Delay from CPU to the inverting buffer in the OE path
Toe2	Inverting buffer delay
Toe2	Delay from the inverting buffer to the SRAM outputs
Twe1	Delay from CPU to the inverting buffer in the WE path
Tweb	Inverting buffer delay
Twe2	Delay from the inverting buffer to the SRAM inputs

**Table B–3 Typical SRAM Specifications**

Function	Description
Toe	Access from OE valid to data valid
Twc	Write cycle time
Twp	Write pulse width
Tdw	Data setup to write pulse deassertion
Tdh	Data hold from write pulse deassertion
Taw	Address setup to write pulse deassertion
Twr	Address hold from write pulse deassertion
Tas	Address setup to write pulse assertion

**Table B–4 CPU Specifications**

Function	Description
Taod	<b>data_ram_oe_h</b> output delay.
Taoh	<b>data_ram_oe_h</b> output hold time.
Tdd	Maximum driver delay with 10-pF load (1.6 ns)
Tdod	Data output delay = Tdd + 0.4 ns skew.

(continued on next page)

## B.4 CPU Bus Interface Timing

**Table B-4 (Cont.) CPU Specifications**

Function	Description
Tdsu	Amount of time required by the 21164 for the data to be ready and stable before the CPU latches it on the next rising edge of a cycle (internal CPU setup time).
Tiod	Index delay = Tdd + 0.4 ns skew.

## B.5 Bcache Read and Write Timing Calculations

### B.5 Bcache Read and Write Timing Calculations

The following sections describe methods of calculating read and write cycle times.

#### B.5.1 Read Cycle Calculation

In the 21164, after a Bcache read command begins on CPU cycle N, at time T, there will be a driver delay T<sub>dd</sub> that will slow down the signal and cause it to appear at the pins at time T + T<sub>dd</sub>. There will also be some clock skew of 0.4 ns, delaying some signals until time T + T<sub>dd</sub> + 0.4 ns. For the data RAM output enable, this delay is T<sub>aod</sub>. When computing BC\_RD\_SPD, this number should be taken into account as shown in the following equation:

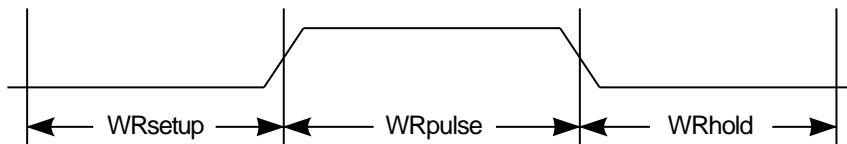
$$\text{Read} = T_{aod} + T_{oe1} + T_{oeb} + T_{oe2} + T_{oe} + T_{dat} + T_{dsu}$$

In other words, the time it takes to perform a Bcache read is the sum of the times it takes the OE signals to travel out of the CPU (T<sub>aod</sub>), through the buffers to the cache SRAMs (T<sub>oe2</sub>), plus the time to read the SRAMs (T<sub>oe</sub>), and the data to get back to the CPU pins (T<sub>dat</sub>), plus the data setup time required by 21164 (T<sub>dsu</sub>).

#### B.5.2 Write Cycle Calculations

WR<sub>setup</sub> is the earliest from the beginning of a write cycle that the write pulse can be asserted (see Figure B-1).

Figure B-1 Write Cycle Timing



MK-2306-13

$$\text{WR}_{\text{setup}} = T_{\text{adr}} + \text{skew}$$

The board-level skew is added to account for any timing differences that may be caused by components in the path, such as the threshold differences inside the SRAM array.

## B.5 Bcache Read and Write Timing Calculations

Once the index signals have reached the SRAMs and the write-enable has been asserted, it must be determined when the write-enable signal can be deasserted. This is done by computing how much time it takes to perform the write operation, assuming the data is already present. This is  $WR_{setup} + T_{aw}$ , where  $T_{aw}$  is the amount of time required by the SRAM to write the data. This quantity is called  $T_{address}$ :

$$T_{address} = WR_{setup} + T_{aw}$$

However, there is more than just  $T_{address}$ ; it must be determined how long it takes the data to reach the SRAMs, and how long it needs to be stable before the write-enable can be deasserted ( $T_{dw}$ ):

$$T_{data} = T_{dod} + T_{clock} + T_{dat} + T_{dw}$$

$T_{clock}$  is equal to one CPU cycle. Because the 21164 does not send the data to be written until 1 cycle after the write was started, it must be included in the time before deassertion.

Therefore,  $T_{address}$  and  $T_{data}$  both determine the earliest from the beginning of a write cycle that the write pulse can be deasserted. The greater of these two determine the more critical path upon which the write pulse is determined.

Because  $WR_{setup}$  is the earliest that the write pulse can be asserted, and  $T_{address}$  and  $T_{data}$  determine the earliest that the write pulse can be deasserted, it follows that  $WR_{pulse} = \text{Maximum of the following:}$

$$\begin{aligned} T_{address} - WR_{setup} &= T_{aw} \\ T_{data} - WR_{setup} \\ T_{wp} + \text{skew} \end{aligned}$$

Once the CPU has deasserted the write-enable signal, it takes some time for it to reach the SRAMs (write-enable path delay). Therefore, that time must be taken into account:

$$WR_{hold} = T_{we1} + T_{buf} + T_{we2} + \text{skew}$$

The  $WR_{setup}$  requirement is offset by the write-enable path delay ( $WR_{hold}$ ) and the  $WR_{hold}$  requirement is offset by the address path delay ( $WR_{setup}$ ). The  $WR_{setup}$  and  $WR_{hold}$  delays are each then discounted by the fastest possible delay through the other path. The minimum parameters estimate the absolute fastest propagation through the address and write-enable paths.

Therefore:

$$\begin{aligned} WR_{setup} &= T_{adr} + \text{skew} - WR_{hold}(\text{minimum}) \\ WR_{hold} &= T_{we1} + T_{buf} + T_{we2} + \text{skew} - WR_{setup}(\text{minimum}) \end{aligned}$$

## B.5 Bcache Read and Write Timing Calculations

### B.5.3 Read/Write Spacing Calculations

The 21164 uses the RD\_WR\_SPC field as the number of CPU cycles to insert between a private read operation followed by a private write operation. The number should be large enough to allow the Bcache drivers to turn off before the 21164 data drivers are turned on, thus avoiding a data bus clash.

To compute this value, the worst case delay of **data\_ram\_oe\_h** signal needs to be determined. The value is given by the following equation:

$$\text{Data\_OE\_delay} = \text{Taoh} + \text{Tdoe1} + \text{Tbuf} + \text{Tdoe2} + \text{Thoz}(\text{Tristate\_RAM\_Turnoff})$$

## B.6 Memory Initialization

Eight consecutive row address strobe (RAS) cycles are performed to the system memory bank to “wake up” the DRAMs. This is done by reading the bank eight times. The caches are disabled at this point so the read data goes directly to the DRAMs (except for the Scache, which cannot be turned off).

Good data parity is ensured by writing all memory locations. This is done by rewriting the full contents of memory with the same data. Reading before writing memory lengthens the time to initialize data parity, however, it conserves the memory state for debugging purposes.

## B.7 Bcache Initialization

The Bcache is initialized by the following steps:

1. Set the BC\_CONTROL register in the CPU to ignore parity/ECC reporting.
2. Turn on the Bcache in the 21164 and the CIA.
3. Sweep the Bcache with read operations at cache block increments.
4. Reenable error reporting.
5. Clear error registers.

When the system is powered up, the Bcache contains UNPREDICTABLE data in the tag RAMs. As the Bcache is swept for initialization, the old blocks (referred to as dirty victim blocks) are written back to main memory. These victim write operations will occur based on the tag address (tag), which stores the upper part of the address location for the dirty blocks of memory.

Because the tags are unpredictable, the victim write operations could occur to UNPREDICTABLE addresses. Therefore, these write operations could be attempted to nonexistent memory. Should this happen, the transaction would complete and an error will be reported. Therefore, reporting of all nonexistent memory errors in the CIA must be turned off prior to sweeping.

## B.8 Special ROM Header

### B.8 Special ROM Header

The MAKEROM tool is used to place a special header on ROM image files. The SROM allows the system (flash) ROM to contain several different ROM images, each with its own header. The header informs the SROM where to load the image, and whether or not it has been compressed with the MAKEROM tool. The header is optional for system ROMs containing a single image. If the header does not exist, the complete 1MB system flash ROM is loaded and executed starting at physical address zero. Figure B-2 shows the header content.

Figure B-2 Special Header Content

31	0
Validation Pattern 5A5AC3C3	0x00
Inverse Validation Pattern A5A53C3C	0x04
Header Size (Bytes)	0x08
Image Checksum	0x0C
Image Size (Memory Footprint)	0x10
Decompression Flag	0x14
Destination Address Lower Longword	0x18
Destination Address Upper Longword	0x1C
Reserved<31:16>   Firmware ID<15:8>   Header Rev<7:0>	0x20
Flash ROM Image Size	0x24
Optional Firmware ID<31:0>	0x28
Optional Firmware ID<63:32>	0x2C
Header Checksum (excluding this field)	0x30

MK-2306-19

## B.8 Special ROM Header

Table B–5 describes each entry in the special header.

**Table B–5 Special Header Entry Descriptions**

Entry	Description
Validation and inverse validation pattern	This quadword contains a special signature pattern used to validate that the special ROM header has been located. The pattern is 5A5AC3C3A5A53C3C.
Header size (bytes)	This longword provides the size of the header block, which varies among versions of the header specification.  When the header is located, SROM code determines where the image begins based on the header size. Additional data added to the header is ignored by older SROM code.  A header size of 32 bytes implies version 0 of the header specifications.
Image checksum	This longword is used to verify the integrity of the ROM.
Image size	The image size is used by the SROM code to determine how much of the system flash ROM should be loaded.
Decompression flag	The decompression flag informs the SROM code whether the MAKEROM tool was used to compress the ROM image with a repeating byte algorithm. The SROM code contains routines that execute the decompression algorithm. Other compression and decompression schemes, which work independently from this scheme, may be employed.
Destination address	This quadword contains the destination address for the image. The SROM code loads the image at this address and begins execution.
Header revision	The revision of the header specification used in this header. This is necessary to provide for changes to the header specification. Version 0 headers are identified by the size of the header (32 bytes).
Flash ROM image size	The flash ROM image size reflects the size of the image as it is contained in the flash ROM.

(continued on next page)

## B.8 Special ROM Header

**Table B-5 (Cont.) Special Header Entry Descriptions**

Entry	Description												
Firmware ID	The firmware ID is a byte that specifies the firmware type. This information facilitates image boot options necessary to boot different operating systems.												
	<table border="1"><thead><tr><th>Firmware Name</th><th>Firmware Type</th><th>Firmware Description</th></tr></thead><tbody><tr><td>Debug monitor</td><td>0</td><td>Alpha evaluation board debug monitor</td></tr><tr><td>Windows NT</td><td>1</td><td>Windows NT ARC firmware</td></tr><tr><td>Alpha SRM</td><td>2</td><td>Alpha System Reference Manual console</td></tr></tbody></table>	Firmware Name	Firmware Type	Firmware Description	Debug monitor	0	Alpha evaluation board debug monitor	Windows NT	1	Windows NT ARC firmware	Alpha SRM	2	Alpha System Reference Manual console
Firmware Name	Firmware Type	Firmware Description											
Debug monitor	0	Alpha evaluation board debug monitor											
Windows NT	1	Windows NT ARC firmware											
Alpha SRM	2	Alpha System Reference Manual console											
Optional firmware ID	This optional field can be used to provide additional firmware information such as firmware revision or a character descriptive string of up to 8 characters.												
Header checksum	The checksum of the header. This is used to validate the presence of a header beyond the validation provided by the validation pattern.												



## B.9 Flash ROM Structure

### B.9 Flash ROM Structure

During the power-up and initialization sequence, the Alpha PCI 164-266/300 always loads the first image if `BOOT_OPTION=1` (jumper J1—25/26 not installed). Then the first image (the debug monitor) will be booted.

If jumper J1—25/26 (`BOOT_OPTION`) is installed (see Figure 2–2), the Alpha PCI 164-266/300 reads the value at location 0x3F of the TOY RAM. The Alpha PCI 164-266/300 uses the value found there to determine which image will be selected (see Table B–6). The selected image is loaded and executed.

**Table B–6 Flash ROM Image Selection**

TOY RAM Value <sup>1</sup>	Firmware ID <sup>2</sup>	Image Description
0x00	0	Evaluation board debug monitor firmware
0x01	1	Windows NT ARC firmware
0x02	2	Alpha SRM firmware (OpenVMS) <sup>3</sup>
0x03	2	Alpha SRM firmware (Digital UNIX) <sup>4</sup>
0x8 <i>n</i>	Not applicable	SROM code loads the <i>n</i> th image from flash ROM. If <i>n</i> =0, the SROM code loads the entire flash ROM contents. If <i>n</i> =1, 2, . . . , the SROM code loads the first image, second image, and so on.

<sup>1</sup>Operating system type. Found at TOY RAM address 0x3F.

<sup>2</sup>Found in image header.

<sup>3</sup>The Alpha PCI 164-266/300 does not support OpenVMS.

<sup>4</sup>The current version of the Alpha PCI 164-266/300 does not support Digital UNIX.

If an image is specified and is not found, the Alpha PCI 164-266/300 loads the first image found in the flash ROM with a valid header. If no valid header is found, the entire 1MB flash image is loaded at address 0x00000000.

The following sequence of steps describes how to change the value stored in TOY RAM location 0x3F using either the basic debug monitor commands or the debug monitor `bootopt` command.

## B.9 Flash ROM Structure

### Changing TOY RAM Location 3F—Basic Debug Monitor Commands

Follow this procedure to change the value in location 0x3F, then load and start the selected image:

1. Remove the jumper at J1—25/26 (BOOT\_OPTION) (see Figure 2-2).
2. Turn power on. The debug monitor will be loaded.
3. Determine the TOY RAM value for the image you have chosen.
4. Use the debug monitor to write *nn* (selected operating system type from Table B-6) in TOY RAM location 0x3F. Check that the value *nn* was written correctly by reading *nn* from the location.

```
> wb 70 3F
> wb 71 nn
> wb 70 3F
> rb 71
```

```
nn
```

5. Turn power off.
6. Install the jumper at J1—25/26.
7. Turn power on. The Alpha PCI 164-266/300 will load and execute the image specified by the value in TOY RAM 0x3F.

## B.9 Flash ROM Structure

### Changing TOY RAM Location 0x3F—Debug Monitor `bootopt` Command

Use the debug monitor `bootopt` command to change the value in location 3F. In the example shown here, the `bootopt` command is used to change the value in location 3F from 0 to 1 to boot Windows NT:<sup>1</sup>

```
EB164> bootopt ❶
Predefined bootoptions are...
"0" "Alpha Evaluation Board Debug Monitor" "DBM"
"1" "The Windows NT Operating System" "NT"
"2" "OpenVMS" "VMS"
"3" "Digital UNIX" "UNIX"
O/S type selected: "Alpha Evaluation Board Debug Monitor"
...Firmware type: "DBM Firmware"
EB164> bootopt nt ❷
O/S type selected: "The Windows NT Operating System"
...Firmware type: "Windows NT Firmware"
EB164> bootopt ❸
Predefined bootoptions are...
"0" "Alpha Evaluation Board Debug Monitor" "DBM"
"1" "The Windows NT Operating System" "NT"
"2" "OpenVMS" "VMS"
"3" "Digital UNIX" "UNIX"
O/S type selected: "The Windows NT Operating System"
...Firmware type: "Windows NT Firmware"
EB164>
```

- ❶ Use the debug monitor `bootopt` command to see the image choices and note which image is selected.
- ❷ Use the debug monitor `bootopt nt` command to change the selected image from 0 to 1.
- ❸ Use the debug monitor `bootopt` command to verify that the selected image has changed from 0 to 1.

<sup>1</sup> The Alpha PCI 164-266/300 uses the same firmware as the Alpha EB164, so the prompts appear as EB164>.

## B.10 Flash ROM Access

### B.10 Flash ROM Access

The flash ROM can be viewed as two banks of 512KB each. At power-up the lower 512KB bank is accessed using the address range 86.FFF8.0000 to 86.FFFF.FFFF.

Setting address bit 19 (**flash\_adr19**) allows you to access the higher 512KB of flash ROM. Write a 1 to the register at address 0x800 to set address bit 19. Manually deposit a 1 to address 0x800 or enter the following command from the debug monitor:

```
> wb 800 1
```

The address range for the higher bank is 86.FFF8.0000 to 86.FFFF.FFFF, the same as for the lower bank. Access is now to the higher bank and will continue until the Alpha PCI 164-266/300 is reset, or a 0 is written to the register at address 0x800.

---

#### Note

---

The write-enable jumper must be installed at J14—2/3 (see Figure 2-1 and Figure 2-2). This enables writing to the flash ROM.

---

## B.11 Icache Flush Code

### B.11 Icache Flush Code

The following code is loaded into memory after the system ROM image. The code is then executed to flush the SROM initialization code from the Icache. The SROM initialization code is loaded into the Icache and maps to memory beginning at address zero.

```
77FF0119 mt    r31, flushIc
C0000001 br    r0, +4
        .long destination
6C008000 ldl_p r0, 0x0 (r0)
47FF041F bis   r31, 31, 31
47FF041F bis   r31, 31, 31
        .
        . (total of 44 bis instructions)
        .
47FF041F bis   r31, 31, 31
47FF041F bis   r31, 31, 31
6BE00000 jmp   r31, (r0)
```

In an attempt to transfer execution to the first page in memory, execution would continue in the SROM initialization code at that address. Therefore, execution must be transferred to some address that does not hit in the Icache where other code can flush the Icache.

The NOPs following the Icache flush allow the instructions that were fetched before the Icache was updated to be cleared from the pipeline. Execution will ultimately continue at the address contained in r0. At this point r0 contains the starting address where the system flash ROM image was loaded into memory.



# C

---

## Technical Support and Additional Documentation

### C.1 Technical Support

If you need technical support with your Alpha PCI Motherboard, contact your local Digital representative. Please provide your local representative with the model number and if possible a brief description of the problem you are encountering.

Additional technical documentation is available from Digital on the major Digital semiconductor components used on your PCI Motherboard. A complete list of these documents can be obtained from your local representative.

And be sure to visit Digital Equipment's home page at UIC:

<http://www.digital.com>

Select the Semiconductor InfoCenter for pointers to relevant technical documentation.

## C.2 Ordering Associated Literature

### C.2 Ordering Associated Literature

The following table lists some of the literature available from Digital. For a complete list, contact your Digital representative.

Title	Order Number
Alpha Architecture Reference Manual <sup>1</sup>	EY-T132E-DP-YCH
Alpha AXP Architecture Handbook	EC-QD2KA-TE
Alpha 21164 Microprocessor Data Sheet	EC-QAEP-TE
Alpha 21164 Microprocessor Hardware Reference Manual	EC-QAEOC-TE
Alpha 21164 Microprocessor Product Brief	EC-QAENB-TE
DECchip 21171 Core Logic Chipset Product Brief	EC-QC3EB-TE
DECchip 21171 Core Logic Chipset Technical Reference Manual	EC-QE18B-TE
Answers to Common Questions about PALcode for Alpha AXP Systems	EC-N0647-72
PALcode for Alpha Microprocessors System Design Guide	EC-QFGLB-TE
Alpha PCI 164-266/300 documentation set	EK-AL266-DK
<i>Alpha Microprocessors SROM Mini-Debugger User's Guide</i>	
<i>Alpha Microprocessors Motherboard Debug Monitor User's Guide</i>	
<i>Alpha PCI 164-266/300 Design Guide</i>	
<i>Alpha PCI 164-266/300 User's Manual</i>	
<i>Alpha PCI 164-266/300 Microsoft Windows NT Installation Guide</i>	

<sup>1</sup>To order and purchase the *Alpha Architecture Reference Manual*, call **1-800-DIGITAL** from the U.S. or Canada, or contact your local Digital office, or technical or reference bookstore where Digital Press books are distributed by Prentice Hall.



## C.3 Ordering Third-Party Literature

### C.3 Ordering Third-Party Literature

You can order the following third-party literature directly from the vendor:

Title	Vendor
PCI System Design Guide	PCI Special Interest Group 1-800-433-5177 (U.S.) 1-503-797-4207 (International) 1-503-234-6762 (FAX) <a href="http://www.teleport.com/~pc2/pcisigindex.html">http://www.teleport.com/~pc2/pcisigindex.html</a>
PCI Local Bus Specification, Revision 2.1	See previous entry.
82420/82430 PCIset ISA and EISA Bridges (includes 82378IB/ZB SIO) PN 290483	Intel Corporation Literature Sales P.O. Box 7641 Mt. Prospect, IL 60056 USA 1-800-628-8686 FaxBACK® Service 1-800-628-2283 BBS 1-916-356-3600 <a href="http://www.intel.com">http://www.intel.com</a>
UPI-41AH/42AH Universal Peripheral Interface 8-Bit Slave Microcontroller PN 210393	Intel Corporation (See previous entry.)
Flash Memory PN 210830	Intel Corporation (See previous entry.)
PC87311/PC87312 (SuperI/O II/III) Floppy Disk Controller with Dual UARTs, Parallel Port, and IDE Interface PN 11362	National Semiconductor Corporation 2900 Semiconductor Drive P.O. Box 58090 Santa Clara, CA 95052 USA 1-800-272-9959 <a href="http://www.national.com">http://www.national.com</a>



---

## Glossary

This glossary provides definitions for terms and acronyms associated with the Alpha PCI 164-266/300 and chips, specifically as applied to Alpha architecture.

### **ASIC**

Application-specific integrated circuit.

### **Bcache**

Backup cache. On the Alpha PCI 164-266/300, a board-level L3 cache with a size of between 2MB and 8MB.

### **BIOS**

Basic input-output system. A set of programs encoded in read-only memory (ROM). These programs facilitate the transfer of data and control instructions between the computer and peripherals, such as, ISA devices and the keyboard.

### **bridge**

The circuitry that connects one computer bus to another computer bus and allows an agent on one bus to access an agent on the other bus.

### **bus**

A group of signals that consists of many transmission lines or wires. It interconnects computer system components to provide communications paths for addresses, data, and control information. The buses used in the Alpha PCI 164-266/300 include PCI64, PCI32, and ISA.

### **cache memory**

A small, high-speed memory placed between slower main memory and the processor. A cache increases effective memory transfer rates and processor speed. It contains copies of data recently used by the processor and fetches several bytes of data from memory, anticipating that the processor will access the next sequential series of bytes. The Alpha 21164 microprocessor contains three onchip internal caches, one 8KB L1 cache for instructions, one 8KB L1

cache for data, and one unified 96KB L2 combined instruction and data cache. *See also* Bcache and write-back cache.

### **CAS**

Column address strobe.

### **CIA**

Control, I/O interface, and address chip. Part of the DECchip 21171 chipset.

### **CMOS**

Complementary metal-oxide semiconductor.

### **Dcache**

Data cache. An 8KB L1 cache reserved for data on the 21164 chip.

### **DRAM**

Dynamic random-access memory. Read/write memory that must be refreshed (read from or written to) periodically to maintain the storage of information.

### **DSW**

Data switch chip. Part of the DECchip 21171 chipset.

### **ECC**

Error correction code. A 16-bit ECC is passed on the 21164's **data\_check** lines for each 128-bit data transfer.

### **flash ROM**

Flash read-only memory. On the Alpha PCI 164-266/300, a 1MB, nonvolatile, writable ROM.

### **Icache**

Instruction cache. An 8KB L1 cache reserved for instructions on the 21164 chip.

### **IPR**

Internal processor register.

### **ISA**

Industry Standard Architecture. An 8-bit or 16-bit interface for interconnecting data storage, data processing, and peripheral control devices in a closely coupled configuration.

**local bus**

A bus that is in close proximity to the CPU and shares its speed. PCI is a local bus.

**PAL**

Programmable array logic.

**PCI**

Peripheral component interconnect. The 64-bit and 32-bit local bus developed by Intel.

**PGA**

Pin grid array.

**PLA**

Programmable logic array.

**PLD**

Programmable logic device.

**PLL**

Phase-locked loop.

**PPGA**

Plastic pin grid array.

**PQFP**

Plastic quad flat pack.

**primary cache**

The cache that is the fastest and closest to the processor. The Alpha 21164 microprocessor contains instruction, data, and unified instruction and data caches. *Also called L1 cache.*

**RAM**

Random-access memory.

**RAS**

Row address strobe.

**region**

One of four areas in physical memory space based on the two most significant, implemented, physical address bits.

**RISC**

Reduced instruction set computing. A computing system architecture with an instruction set that is paired down and reduced in complexity so that most instructions can be performed in a single processor cycle. High-level compilers synthesize the more complex, least frequently used instructions by breaking them down into simpler instructions. This approach allows the RISC architecture to implement a small, hardware-assisted instruction set, thus eliminating the need for microcode.

**Scache**

Secondary cache. A 96KB L2 cache reserved for instructions and data within the 21164 chip.

**SCSI**

Small computer system interface. An interface standard for peripheral devices such as hard disk drives, CD-ROM drives, and tape drives. The drive contains most of the controller circuitry, leaving the SCSI interface free to communicate with other peripherals.

**SIMM**

Single inline memory module.

**SRAM**

Static random-access memory.

**SRAM**

Serial read-only memory.

**UART**

Universal asynchronous receiver-transmitter.

**word**

Two contiguous bytes (16 bits) starting on an arbitrary byte boundary. The bits are numbered from right to left, 0 through 15.

**write-back cache**

A cache in which copies are kept of any data in the region. Read and write operations may use the copies, and write operations use additional states to determine whether there are other copies to invalidate or update.

**write-through cache**

A cache in which copies are kept of any data in the region. Read operations may use the copies, but write operations update the actual data location and either update or invalidate all copies.





---

# Index

21171

*See* CIA chip or DSW chip

64-byte mode, 3-3

## A

---

Address map

bridge

configuration registers, A-16, A-17

operating registers, A-8, A-9

PC87312 registers, A-2

physical, A-1

SIO

configuration registers, A-16, A-17

operating registers, A-8, A-9

Address space

PCI

configuration, A-16

dense memory, A-13

interrupt acknowledge/special cycle,  
A-18

sparse I/O, A-1

sparse memory, A-1

Airflow requirements, 5-2

Associated literature

Alpha documentation, C-2

third-party documentation, C-3

## B

---

Backup cache

*See* Bcache

Bcache, 3-2 to 3-3

configurations, 3-3

index jumpers, 2-5

Bcache (cont'd)

SIMM connectors, 2-11

size, 3-3

size jumpers, 2-4

speed, 3-3

speed jumpers, 2-5

subsystem, 1-3

BC\_RD\_FAST jumper, 2-7

Block diagram, 1-1

Bcache array logic, 3-2

interrupt logic, 3-14

ISA devices, 3-9

main memory, 3-4

power distribution, 3-23

reset/init logic, 3-20

SROM serial port logic, 3-22

system clocks and distribution logic, 3-17

Board

configuration, 2-1

configuration jumpers, 2-1

connectors, 2-8

jumpers, 2-1

uses, 1-6

Board components and features, 1-1

Board configuration jumpers, 2-1

Boot option jumper, 2-6

Bridge

*See* SIO chip

## C

---

### Cache

*See also* Bcache

SIMM connectors, 2-11

### Chipset overview, 3-4

### Chipset support, 1-3

### CIA chip, 3-4

### Clock

frequency multiplier, 3-17

### Clock divisor jumper, 2-4

### Clocks

diskette, 3-19

ISA, 3-19

system, 3-17

### COM1/2 connectors, 2-12

### Combination controller, 3-9

### Components and features, 1-1

### Configuration, 2-1 to 2-16

### Configuration jumpers, 2-1

### Connectors, 2-8

Bcache SIMM, 2-11

debug monitor interface, 2-12

diskette, 2-12

DRAM SIMM, 2-11

fan power, 2-12, 2-15

fan sensor, 2-15

IDE drive, 2-12

ISA bus, 2-11

keyboard, 2-11

Mini-Debugger, 2-12

mouse, 2-12

parallel port, 2-12

part numbers, 2-15 to 2-16

PCI bus, 2-11

power, 2-14, 2-15

serial interface ports, 2-12

speaker, 2-13

SROM port, 2-12

### Conventions, x to xiii

### Cooling fan

power connectors, 2-12, 2-15

sensor connector, 2-15

### CPU

clock, 3-17

### Current

dc ampere requirements, 5-1

## D

---

### dc power distribution, 3-23

### dc power requirements, 5-1

### Debug and monitor ROM

system support, 1-5

### Debug monitor

code, 3-25, 3-26

interface connector, 2-12

### DECchip 21171-BA

*See* DSW chip

### DECchip 21171-CA

*See* CIA chip

### DECchip 21171 chipset, 1-3, 3-4

*See also* CIA chip *See also* DSW chip

### Diskette drive

clock, 3-19

connector, 2-12

### Documentation, C-2, C-3

### DRAM

SIMM connectors, 2-11

### DSW chip, 3-6

### Dynamic RAM

*See* DRAM

## E

---

### Environmental requirements, 5-2

## F

---

### Fan

power connectors, 2-12, 2-15

sensor, 5-1

sensor connector, 2-15

### Flash ROM, 1-4, 1-5, 3-13

Flash ROM write-protect/write-enable jumper, 2-7  
Floppy drive  
  *See* Diskette drive  
Functional description, 3-1 to 3-26  
  Bcache, 3-2 to 3-3  
  dc power distribution, 3-23  
  DECchip 21171 chipset, 3-4 to 3-6  
  interrupts, 3-14 to 3-16  
  ISA bus devices, 3-9 to 3-13  
  main memory interface, 3-6 to 3-7  
  PCI devices, 3-7 to 3-8  
  reset and initialization, 3-20  
  software, 3-25 to 3-26  
  SROM, 3-22  
  system clocks, 3-17 to 3-19

---

## H

Halt button, 2-13  
Hard drive connector, 2-12

---

## I

I/O space  
  address map, A-1  
IDE drive  
  active pins, 2-12  
  connector, 2-12  
**IDSEL** pin select, A-16  
Industry Standard Architecture  
  *See* ISA  
Initialization, 3-20  
Integrated drive electronics  
  *See* IDE drive  
Interrupts, 3-14  
INT*nm*, xiii  
ISA  
  bus slots, 2-11  
  clock, 3-19  
  devices, 3-9  
    combination controller, 3-9  
    keyboard controller, 3-12  
    mouse controller, 3-12

## ISA

  devices (cont'd)  
    time-of-year clock, 3-12  
    Ubus memory device, 3-13  
  expansion slots, 3-13  
ISA interface overview, 1-4

---

## J

Jumpers, 2-1  
  Bcache index, 2-5  
  Bcache size, 2-4  
  Bcache speed, 2-5  
  BC\_RD\_FAST, 2-7  
  board configuration, 2-1  
  boot option, 2-6  
  Flash ROM write-protect/write-enable, 2-7  
  Mini-Debugger, 2-6  
  system clock divisor, 2-4

---

## K

Keyboard  
  connector, 2-11  
  lock switch, 2-13

---

## L

Literature  
  Alpha documentation, C-2  
  third-party documentation, C-3

---

## M

Main memory interface, 3-6  
Memory  
  SIMM connectors, 2-11  
Memory subsystem, 1-3  
Mini-Debugger  
  code, 3-25  
  connector, 2-12  
  jumper, 2-6  
Motherboard uses, 1-6

Mouse connector, 2-12

## O

---

Operating systems, 3-25  
software support, 1-5

OS

*See* Operating systems

## P

---

Parallel port connector, 2-12

PC87312 register address map, A-2

PCI

bus slots, 2-11

configuration address space, A-16

dense memory space, A-13

devices, 3-7

SIO chip, 3-8

expansion slots, 3-8

interface overview, 1-4

interrupt acknowledge/special cycle

address space, A-18

sparse I/O space, A-1

sparse memory space, A-1

Peripheral component interconnect

*See* PCI

Physical board

component layout, 5-2

parameters, 5-2

Ports

SRAM, 3-22

SRAM test, 3-22

Power connectors, 2-14, 2-15

Power distribution, 3-23

Power on indicator, 2-13

Power requirements, 5-1

Power supply

dc ampere requirements, 5-1

wattage requirements, 5-1

## R

---

RAM

*See* DRAM or SRAM

Related documentation, C-2, C-3

Reset, 3-20

button, 2-13

ROM

*See* Flash ROM

## S

---

Saturn I/O chip

*See* SIO chip

Serial interface

connectors, 2-12

Serial ROM

*See* SROM

Serial ROM code

system support, 1-5

SIMM

connectors, 2-11

Single inline memory module

*See* SIMM

SIO chip, 3-8

configuration register address map, A-16,

A-17

operating register address map, A-8, A-9

Slots

ISA bus, 2-11

PCI bus, 2-11

Software, 3-25

Software support, 1-5

Speaker connection pins, 2-13

SRAM, 3-3

SROM, 3-22

code, 3-25

connector, 2-12

functions, 3-22

multiplexer selection, 3-22

port, 3-22

Static RAM

*See* SRAM

Support chipset, 1–3

System

clocks, 3–17

halt button, 2–13

reset button, 2–13

System I/O chip

*See* SIO chip

System software

software support, 1–5

## **T**

---

Test SROM port, 3–22

Time-of-year clock, 3–12

Timing, 3–17

## **U**

---

Ubus, 3–9

memory device, 3–13

Utility bus

*See* Ubus

## **W**

---

Wave pipelining, 3–3

