COMPAQ

AlphaPC 264DP

User’s Manual

Order Number: EC–RB0BA–TE

Revision/Update Information: This is a new document.

Preliminary

Compaq Computer Corporation

February 1999

The information in this publication is subject to change without notice.

COMPAQ COMPUTER CORPORATION SHALL NOT BE LIABLE FOR TECHNICAL OR EDITORIAL ERRORS OR OMISSIONS CONTAINED HEREIN, NOR FOR INCIDENTAL OR CONSEQUENTIAL DAMAGES RESULTING FROM THE FURNISHING, PERFORMANCE, OR USE OF THIS MATERIAL. THIS INFORMATION IS PROVIDED "AS IS" AND COMPAQ COMPUTER CORPORATION DISCLAIMS ANY WARRANTIES, EXPRESS, IMPLIED OR STATUTORY AND EXPRESSLY DISCLAIMS THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR PARTICULAR PURPOSE, GOOD TITLE AND AGAINST INFRINGEMENT.

COMPAQ, the Compaq logo, the Digital logo, and DIGITAL Registered in United States Patent and Trademark Office.

AlphaPC, DECchip, DECnet, DIGITAL UNIX, OpenVMS, ThinWire, and Tru64 are trademarks of Compaq Computer Corporation.

Microsoft, MS-DOS, Windows, Windows NT, and Windows 95 are registered trademarks of Microsoft Corporation.

Other product names mentioned herein may be the trademarks of their respective owners.

Contents

1 About This Manual

2 Features of the AlphaPC 264DP

3 AlphaPC 264DP Switch Configuration

4 AlphaPC 264DP Connector Pinouts

5 Memory Configuration

6 Firmware Introduction

7 AlphaBIOS

8 Alpha SRM Console Firmware

9 Troubleshooting

A Support, Products, and Documentation

Figures

Tables

1 About This Manual

This manual describes the AlphaPC 264DP, a board for computing systems based on the Alpha 21264 microprocessor and the companion 21272 core logic chipset. It describes both the mainboard's and the daughtercard's features and how to set the configuration switches. This manual helps users to install and populate the AlphaPC 264DP mainboard with memory boards and peripheral cards.

1.1 Manual Conventions and Terminology

The following conventions are used in this manual.

Caution: Cautions indicate potential damage to equipment, software, or data.

Extents: Extents are specified by a single number or a pair of numbers in brackets ([ ]) separated by a colon (:), and are inclusive. For example, bits [7:3] specify an extent including bits 7, 6, 5, 4, and 3. Multiple bit fields are shown as extents.

Italic Type: *Italic* type emphasizes important information and indicates complete titles of documents.

Note: Notes provide additional information about a topic.

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.

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

Signal Names: All signal names are printed in **boldface** type. Signal names that originate in an industry-standard specification, such as PCI or IDE, are printed in the case as found in the specification (usually uppercase). Active low signals have either a pound sign "#" appended, or a "not" overscore bar; for example, **DEVSEL#** and **RESET**.

Manual Conventions and Terminology

Terms: The following terms are used in this manual:

This term...	Refers to...
Microsoft Windows NT installation guide	The Microsoft *Windows NT Workstation Installation Guide* and the *Windows NT Server Installation Guide*
Windows NT	The Microsoft Windows NT Workstation and Windows NT Server operating systems
Alpha SRM console	The Alpha SRM console firmware
AlphaBIOS	The AlphaBIOS firmware
Tru64 UNIX	COMPAQ’S 64-bit UNIX operating system, replacing DIGITAL UNIX.

2

Features of the AlphaPC 264DP

The AlphaPC 264DP system (AlphaPC 264DP) consists of an AlphaPC 264DP mainboard (a planar board, referred to as the mainboard throughout this document), and one or two AlphaPC 264DP daughtercards (daughtercard). The daughtercard consists of the 21264 microprocessor, L2 cache, reset FPGA, and power converters for 2.2 volts and 1.5 volts.

Table 2–1 provides an overview of the AlphaPC 264DP's features.

Table 2–1 AlphaPC 264DP Features

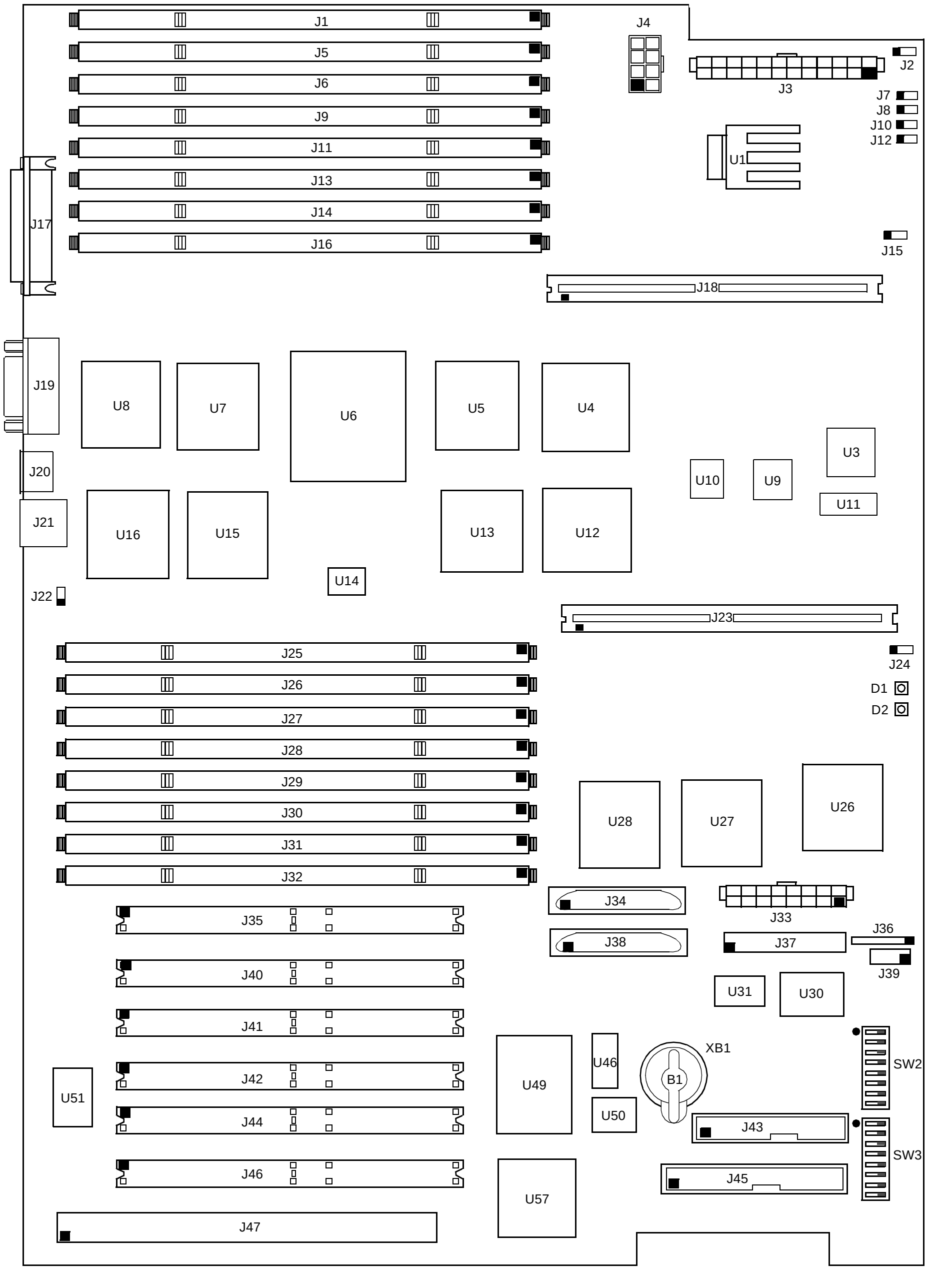
Feature	Description
Mainboard	
Daughtercards	Support for two daughtercards, each containing an Alpha 21264 microprocessor (64-bit RISC) operating at 500 MHz, and a 2MB, 128-bit L2 cache with late-write HSTL synchronous static RAM (SSRAM).
Core logic chipset	21272 core logic chipset, comprising 11 chips (8 Dchips, 2 Pchips, and 1 Cchip) that provide an interface to system memory and the PCI bus.
Synchronous DRAM (SDRAM) memory	128MB to 4GB memory complement, with 72-bit ECC — four memory arrays, each consisting of four 200-pin buffered DIMMs, with qualified 100-MHz SDRAMs.

Table 2–1 AlphaPC 264DP Features (Continued)

Feature	Description
I/O and miscellaneous support	The AlphaPC 264DP mainboard has the following support: • Two 64-bit, 33-MHz PCI buses • Six 64-bit PCI expansion slots (one shared with ISA) • Cypress CY82C693UB chip, providing support for PCI-to-ISA bridge, IDE bus, keyboard, mouse, and time-of-year clock – ISA expansion bus, with one ISA expansion slot (shared with PCI) – IDE bus (Note: Only CD-ROM drives are supported. The maximum supported cable length is 12 inches.) – SMC FDC37C669 combination controller chip provides control for diskettes, two UARTs with modem control, and a parallel port • Adaptec AIC7895 chip, providing integrated dual-port, ultrawide SCSI
Firmware	Windows NT AlphaBIOS and Alpha SRM console.
Daughtercard	
Microprocessor	Alpha 21264 microprocessor (64-bit RISC)
Synchronous L2 cache	Onboard 2MB, direct-mapped, late-write SSRAM cache with 128-bit data path
I/O and miscellaneous support	The AlphaPC 264DP daughtercard has the following support: • 270-pin interface to mainboard • SROM support using 512KB flash ROM • Linear regulator, providing 3.3 V dc to 1.5 V dc conversion • dc-to-dc converter, providing 5 V dc to 2.2 V dc • Reset/configuration FPGA • Presence detect logic, for cache configuration and CPU speed • SROM test port • 18-pin power connector

Figure 2–1 shows the mainboard and its components.

Figure 2–1 AlphaPC 264DP Mainboard Switch/Connector/Component Location



■ indicates pin 1.   ● indicates switch 1.

Table 2–2 describes the mainboard components.

Table 2–2 AlphaPC 264DP Mainboard Switch/Connector/Component List

Item No.	Description	Item No.	Description
XB1	RTC battery (CR2032)	J45	IDE bus connector
J1, J5, J6, J9, J11, J13, J14, J16, J25-J32	Memory connectors	J47	ISA bus connector
J2, J15, J22, J24	Fan box power connector	D1, D2	LEDs
J3	+3-V power connector	SW2, SW3	Switchpacks
J4	Reserved	U1	MIC29502
J7	Power button connector	U3	MC12439
J8	Reset button connector	U4, U5, U7, U8, U12, U13, U15, U16	DC4047 Dchips
J10	SCSI LED connector	U6	DC1046 Cchip
J12	Halt button connector	U9, U10	100LVE222
J17	Parallel I/O connector	U11	MC100LVEL37
J18, J23	Daughtercard connectors	U14	MPC951
J19	COM1/COM2 (DB9) connectors[1]	U26	TIGbus FPGA
J20	Reserved	U27, U28	DC1048 Pchips
J21	Keyboard/mouse connector[2]	U30	AlphaBIOS flash ROM
J33	+5-V power connector	U31	I^{2}C bus controller
J34, J38	SCSI connectors	U46	SRAM for SCSI
J35, J40-J42, J44, J46	PCI bus connectors	U49	AIC7895
J36	Power LED connector	U50	SCSI BIOS flash ROM
J37	Reserved	U51	Super I/O (FDC37C669)
J39	Speaker connector	U57	Southbridge (CY82C693UB)
J43	Floppy drive connector		

[1] COM1 is the top connector, COM2 is the bottom one.
[2] Mouse connector is on the top, keyboard connector is on the bottom.

Figure 2–2 shows the daughtercard and its components, and Table 2–3 describes these components.

Figure 2–2 AlphaPC 264DP Daughtercard Switch/Connector/Component Location

Side 1–Component Side

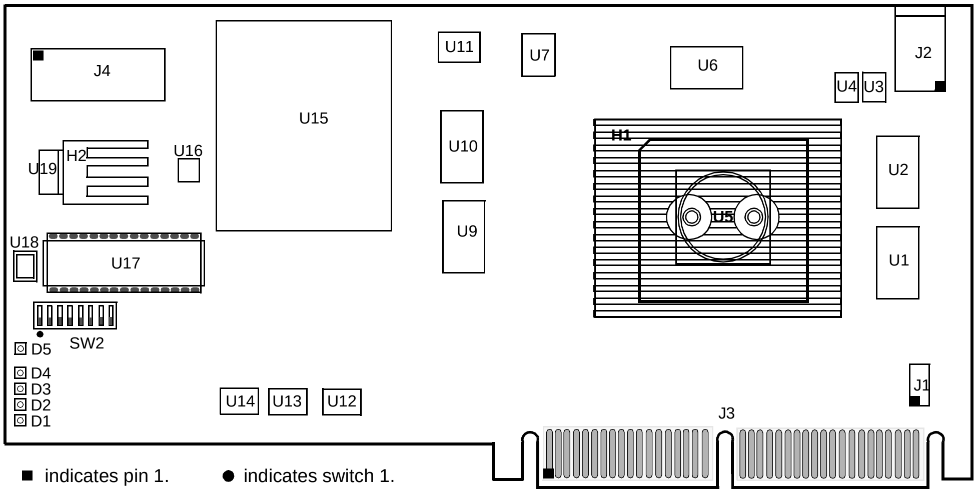


Table 2–3 AlphaPC 264DP Daughtercard Switch/Connector/Component List

Item No.	Description	Item No.	Description
J1	Fan power	U5	Microprocessor, socketed (Alpha 21264)
J2	SROM debug connector	U6	Bcache tag SSRAM
J3	Daughtercard data connector	U7	Reset FPGA
J4	Daughtercard power connector	U11, U13, U14	lcx38
H1	21264 heat sink	U12	8582 EEPROM
H2	+1.5-V regulator heat sink	U15	5-V to 2.2-V converter
D1-D5	LEDs	U16	tl7702b supervisor
SW2	Switchpack	U17	512K×8 flash ROM, socketed
U1, U2, U9, U10	Bcache data SSRAMs	U18	74f151 multiplexer
U3	1489	U19	mic29302 3.3-V to 1.5-V regulator
U4	1488		

2.1 Power Requirements

The mainboard has a maximum total power dissipation of 215 W, excluding any disk drives. Each daughtercard has a maximum total power dissipation of 129 W. Table 2–4 lists the current requirement for each dc supply voltage.

Table 2–4 Power Supply DC Current Requirements

Voltage/Tolerance	Current
Mainboard	
+3.3 V dc, ±5%	30.0 A
+5 V dc, ±5%	20.0 A
5 VSB dc, ±5%	1.0 A
+12 V dc, ±5%	0.8 A
–12 V dc, ±5%	0.1 A
Daughtercard	
+3.3 V dc, ±5%	5.0 A
+5 V dc, ±5%	22.0 A
+12 V dc, ±5%	0.1 A
–12 V dc, ±5%	0.05 A

Caution: **Fan sensor required**. The 21264 microprocessor cooling fan *must* have a built-in sensor that will drive a signal if the airflow stops. The sensor is connected to power connector J1. When the signal is generated, it resets the system.

2.2 Environmental Requirements

The 21264 microprocessor is cooled by a small fan blowing directly into the chip's heat sink. The daughtercard is designed to run efficiently by using only this fan. Additional fans may be necessary depending upon cabinetry and the requirements of plug-in cards.

The mainboard and daughtercard are specified to run within the environment listed in Table 2–5.

Table 2–5 AlphaPC 264DP Environmental Requirements

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 ±2°C/hour (20°F/hour ±4°F/hour)

2.3 Physical Parameters

The AlphaPC 264DP mainboard is a printed-wiring board (PWB) with the following dimensions:

- Length: 42.11 cm (16.58 in ±0.0005 in)
- Width: 33.02 cm (13.0 in ±0.0005 in)
- Height: 3.81 cm (1.5 in)

The AlphaPC 264DP daughtercard is a PWB with the following dimensions:

- Length: 30.48 cm (12.0 in ±0.0005 in)
- Width: 14.99 cm (5.905 in ±0.0005 in)
- Height: 6.40 cm (2.52 in ±0.0005 in)

2.3.1 AlphaPC 264DP Hole and Connector Specifications

Figure 2–3 shows the mainboard’s hole specifications.

Figure 2–3 AlphaPC 264DP Mainboard Hole Specifications

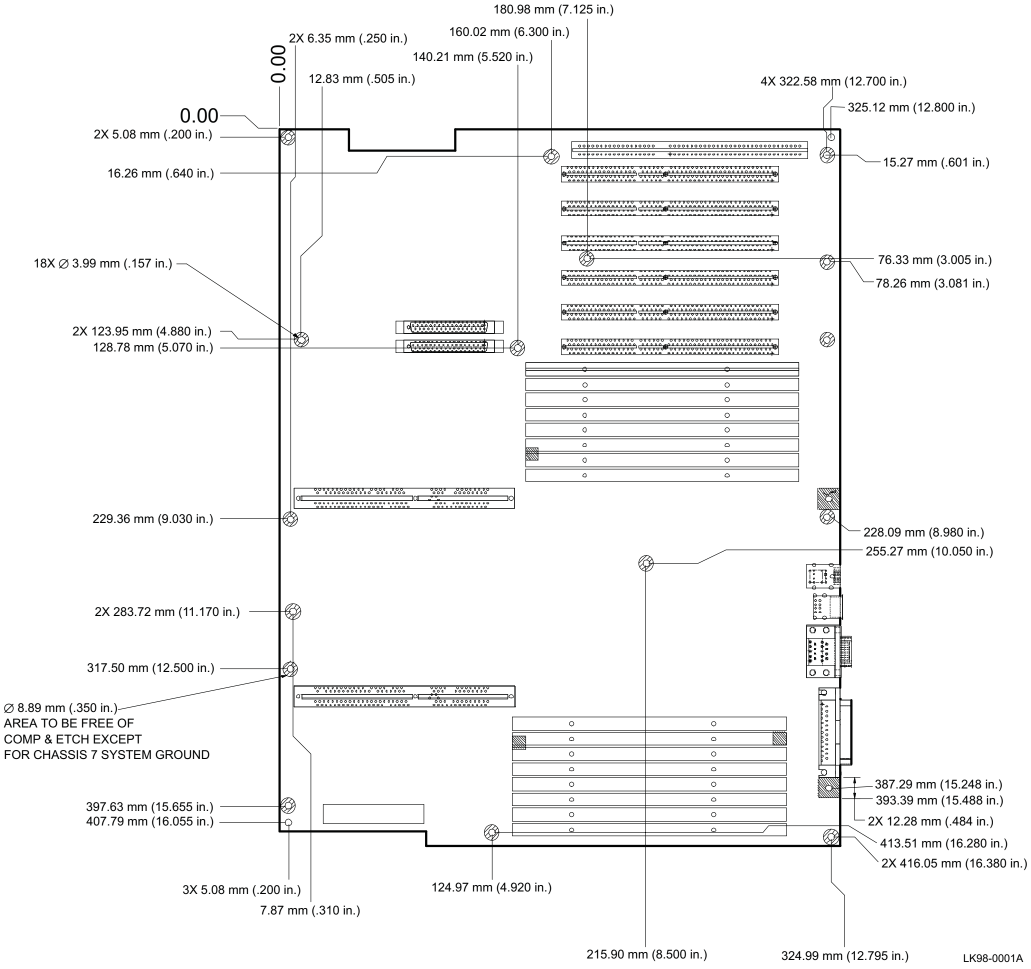
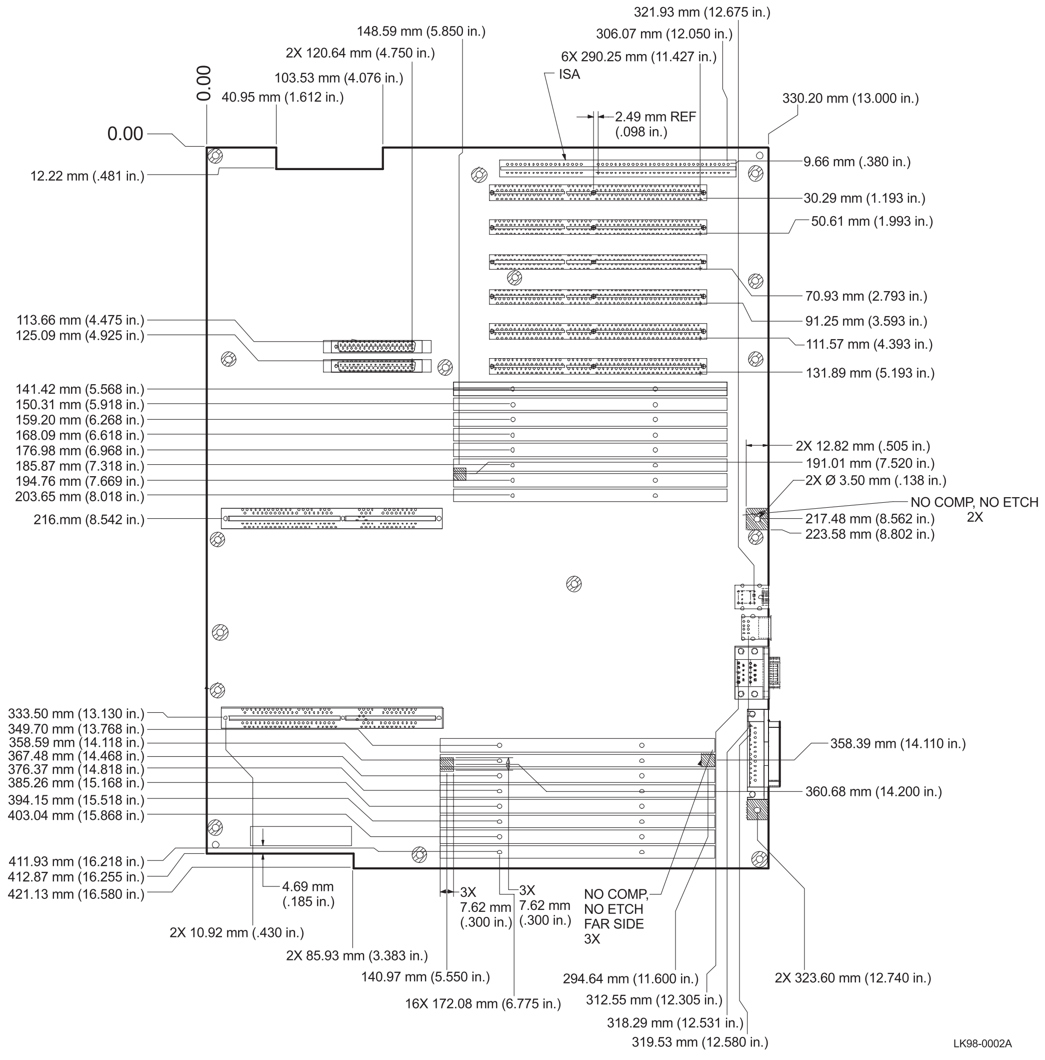


Figure 2–4 shows the mainboard’s connector specifications.

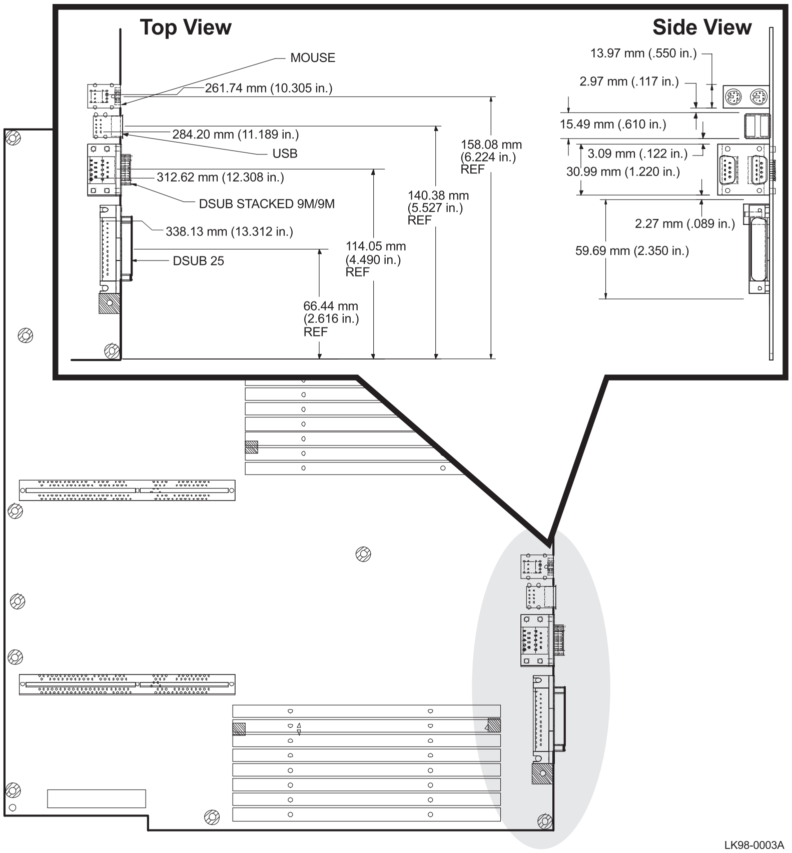
Figure 2–4 AlphaPC 264DP Mainboard Connector Specifications



Physical Parameters

Figure 2–5 shows the top and side views of the mainboard’s I/O connectors.

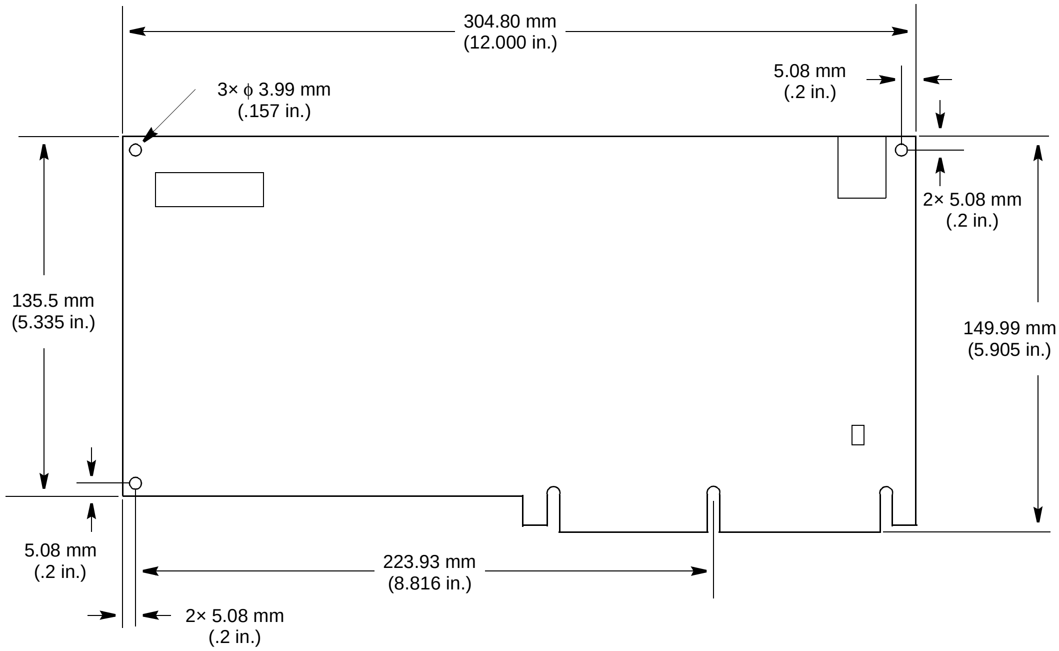
Figure 2–5 AlphaPC 264DP Mainboard I/O Connector Specifications



2.3.2 AlphaPC 264DP Daughtercard Mechanical Control Drawing

Figure 2–6 shows the mechanical control drawing for the AlphaPC 264DP daughtercard.

Figure 2–6 AlphaPC 264DP Daughtercard Mechanical Control Drawing



3

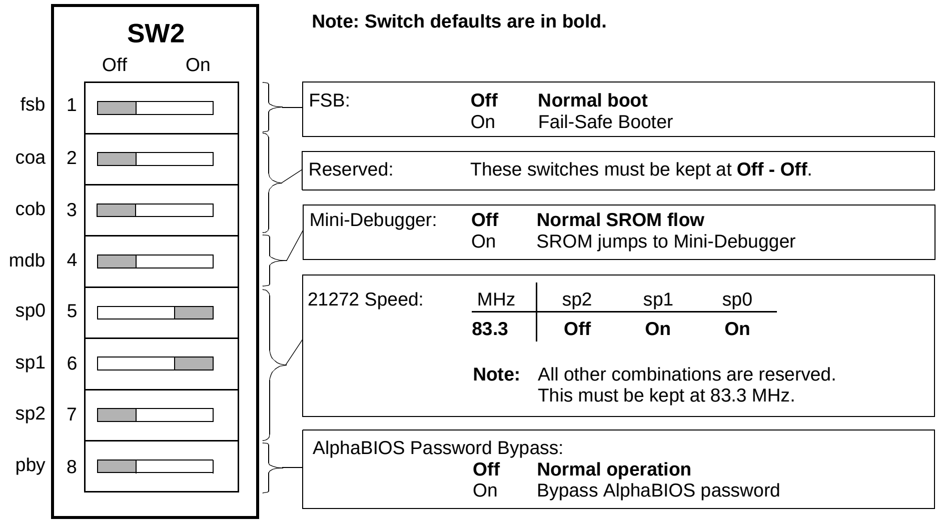
AlphaPC 264DP Switch Configuration

The AlphaPC 264DP mainboard has two sets of programmable switches, located at SW2 and SW3, as shown in Figure 2–1. The AlphaPC 264DP daughtercard has one switchpack, located at SW2, as shown in Figure 2–2. These switches set the hardware configuration.

Note: There are no switchpacks SW1 on production mainboards or daughtercards. Onboard resistors set the configuration (debug use on the mainboard; cache size, CPU speed, and flash ROM use on the daughtercard) to the default state.

Figures 3–1 and 3–2 reflect the mainboard switches, and Figure 3–3 shows the daughtercard switch configurations.

Figure 3–1 Mainboard Switchpack 2



Note: Switch defaults are in bold.

FSB: **Off** **Normal boot**
On Fail-Safe Booter

Reserved: These switches must be kept at **Off - Off**.

Mini-Debugger: **Off** **Normal SROM flow**
On SROM jumps to Mini-Debugger

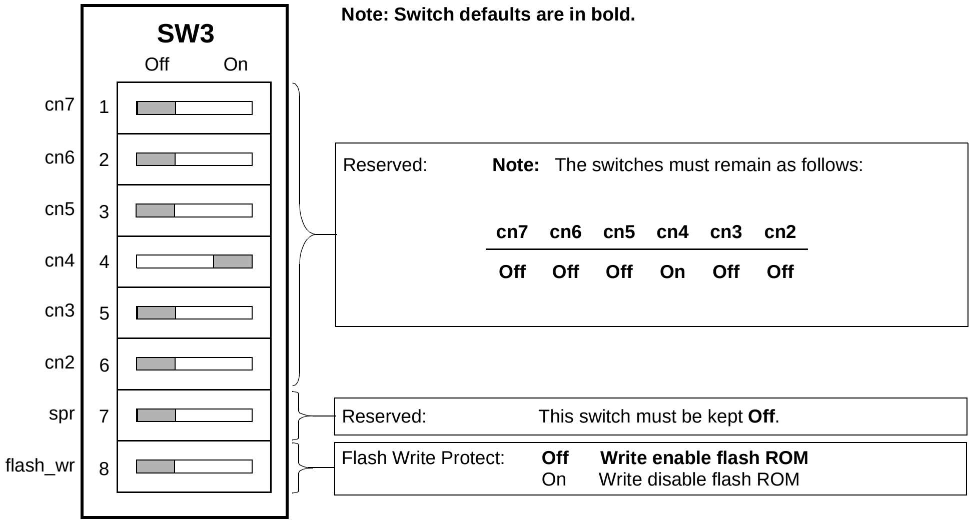
21272 Speed:

MHz	sp2	sp1	sp0
83.3	**Off**	**On**	**On**

Note: All other combinations are reserved.
This must be kept at 83.3 MHz.

AlphaBIOS Password Bypass:
Off **Normal operation**
On Bypass AlphaBIOS password

Figure 3–2 Mainboard Switchpack 3



Note: Switch defaults are in bold.

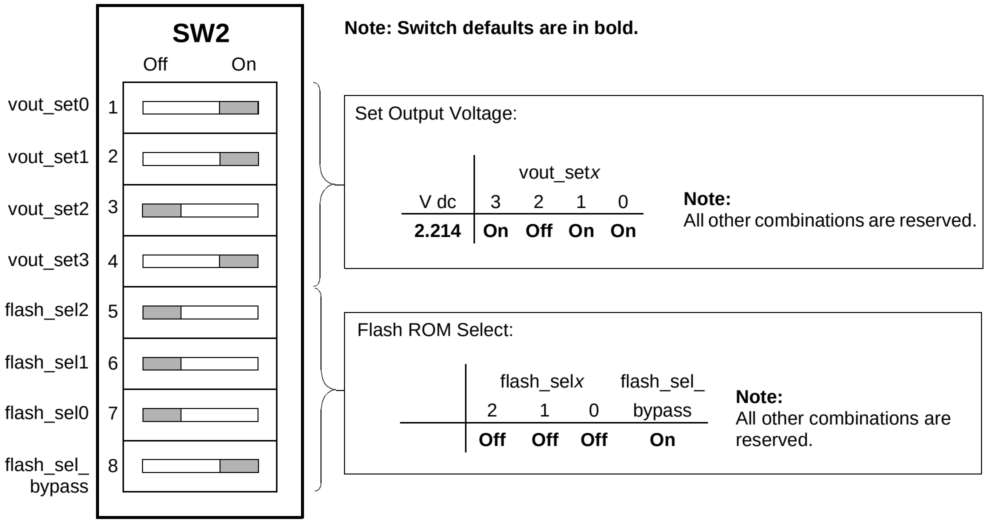
Reserved: **Note:** The switches must remain as follows:

cn7	**cn6**	**cn5**	**cn4**	**cn3**	**cn2**
Off	**Off**	**Off**	**On**	**Off**	**Off**

Reserved: This switch must be kept **Off**.

Flash Write Protect: **Off** **Write enable flash ROM**
On Write disable flash ROM

Figure 3–3 Daughtercard Switchpack 2



Note: Switch defaults are in bold.

Set Output Voltage:

	vout_set*x*			
V dc	3	2	1	0
2.214	**On**	**Off**	**On**	**On**

Note:
All other combinations are reserved.

Flash ROM Select:

flash_sel*x*			flash_sel_
2	1	0	bypass
Off	**Off**	**Off**	**On**

Note:
All other combinations are reserved.

3.1 Fail-Safe Booter

The Fail-Safe Booter (FSB) utility provides an emergency recovery mechanism when the primary firmware image contained in flash memory has been corrupted. When flash memory has been corrupted, and no image can be loaded safely from the flash ROM, you can run the FSB and boot another image from a diskette that is capable of reprogramming the flash ROM. Refer to Chapter 6 for more information.

3.2 Mini-Debugger

The Alpha SROM Mini-Debugger is stored in the flash ROM and is enabled/ disabled by switch 4 of SW2 on the mainboard (see Figure 3–1). The default position for this switch is off. When this switch is on, it causes the SROM initialization to trap to the Mini-Debugger after all initialization is complete, but before starting the execution of the system flash ROM code.

3.3 Password Bypass

AlphaBIOS provides password protection. However, if the use of passwords has been enabled and you have forgotten the current password, password bypass is provided through the use of switch 8 (pby) of SW2 on the mainboard.

Normal operation, with switch 8 in the off position (see Figure 3–1), requires a password. The password bypass function is enabled by setting the switch to the on position. This disables the AlphaBIOS password verification and enables the user to set up or start up their system without the AlphaBIOS password. Password bypass also clears the password.

After this function has been enabled, to disable it and require a password, set switch 8 to the off position.

3.4 Flash Write Protection

The AlphaPC 264DP provides write protection for the firmware flash ROM. By default, writing to the flash ROM is allowed, that is, switch 8 (flash_wr) of SW3 on the mainboard is off (see Figure 3–2). To enable the flash write protection function, set switch 8 to the on position.

Note: The AlphaPC 264DP will not function if switch 8 is in the on position.

3.5 21272 Speed

The speed of the 21272 core logic chipset is determined by switches 5–7 of SW2 on the mainboard. The default positions are 5 and 6 on, 7 off. These switches must be kept in the default position.

4
AlphaPC 264DP Connector Pinouts

This chapter lists the pinouts of the AlphaPC 264DP connectors.

4.1 Mainboard Connectors

Tables 4–1 through 4–19 show the mainboard connector pinouts. See Figure 2–1 for connector locations.

4.1.1 Daughtercard Connector Pinouts

Table 4–1 shows the daughtercard connector pinouts.

Table 4–1 Daughtercard Connector Pinouts (J18, J23)

Pin	Signal	Pin	Signal	Pin	Signal	Pin	Signal
1	**vdd_3v**	2	**vdd_3v**	3	**vdd_3v**	4	**vdd_3v**
5	**vdd_3v**	6	**vdd_3v**	7	**vdd_3v**	8	**vdd_3v**
9	**sysdata2_l**	10	**sysdata0_l**	11	**sysdata5_l**	12	**sysdata3_l**
13	**syscheck0_l**	14	**sysdata9_l**	15	**sysdata10_l**	16	**sysdata13_l**
17	**sysdata14_l**	18	**sysdataoutclk1_l**	19	**sysdata16_l**	20	**syscheck20_l**
21	**sysdata17_l**	22	**sysdata18_l**	23	**sysdataoutclk2_l**	24	**sysdata21_l**
25	**sda**	26	**Gnd**	27	**Gnd**	28	**cpu_slot**
29	**sysdata26_l**	30	**sysdata25_l**	31	**sysdatainclk3_l**	32	**sysdata28_l**
33	**sysdata30_l**	34	**syscheck3_l**	35	**clkfwdreset_h**	36	**irq_0_h**
37	**irq_2_h**	38	**irq_4_h**	39	**2v_pwrgood_h**	40	**tsu_speed0**
41	**tsu_speed1**	42	**clk_rdy_h**	43	**tsu_speed2**	44	**sysfillvalid_l**
45	**sysaddin13_l**	46	**sysaddin8_l**	47	**sysaddin9_l**	48	**sysaddinclk_l**
49	**sysaddout13_l**	50	**sysaddin7_l**	51	**sysaddin1_l**	52	**sysaddin3_l**
53	**sysaddout14_l**	54	**sysaddout12_l**	55	**sysaddout10_l**	56	**sysaddoutclk_l**
57	**sysaddout6_l**	58	**sysaddout0_l**	59	**sysaddout2_l**	60	**syscheck7_l**
61	**sysaddout1_l**	62	**sysdata60_l**	63	**sysdatainclk7_l**	64	**sysdata52_l**

Table 4–1 Daughtercard Connector Pinouts (J18, J23) (Continued)

Pin	Signal	Pin	Signal	Pin	Signal	Pin	Signal
65	**sysdata54_l**	66	**sysdata57_l**	67	**sysdataoutclk6_l**	68	**sysdata51_l**
69	**sysdata49_l**	70	**syscheck5_l**	71	**sysdata46_l**	72	**sysdata47_l**
73	**sysdata45_l**	74	**sysdata42_l**	75	**sysdatainclk5_l**	76	**sysdata41_l**
77	**sysdataoutclk4_l**	78	**sysdata37_l**	79	**sysdata36_l**	80	**sysdata35_l**
81	**sysdata32_l**	82	**Gnd**	83	**bc_config1**	84	**bc_config3**
85	**cpu_speed1**	86	**+12v_mod**	87	**vdd_2v_term**	88	**vdd_2v_term**
89	**vdd_2v_term**	90	**vdd_2v_term**	91	**vdd_3v**	92	**vdd_3v**
93	**vdd_3v**	94	**vdd_3v**	95	**vdd_3v**	96	**vdd_3v**
97	**vdd_3v**	98	**vdd_3v**	99	**sysdatainclk0_l**	100	**sysdata6_l**
101	**sysdata1_l**	102	**sysdataoutclk0_l**	103	**sysdata4_l**	104	**sysdata7_l**
105	**sysdata8_l**	106	**sysdatainclk1_l**	107	**sysdata11_l**	108	**sysdata12_l**
109	**sysdata15_l**	110	**sysdata19_l**	111	**sysdatainclk2_l**	112	**sysdata20_l**
113	**sysdata22_l**	114	**sysdata23_l**	115	**pllbypass_h**	116	**srom_en_l**
117	**sclk**	118	**syscheck2_l**	119	**sysdata24_l**	120	**sysdata27_l**
121	**sysdataoutclk3_l**	122	**sysdata29_l**	123	**sysdata31_l**	124	**sysdataoutvalid_l**
125	**Gnd**	126	**Gnd**	127	**fan_ok_l**	128	**irq_1_h**
129	**irq_3_h**	130	**irq_5_h**	131	**mod_reset_l**	132	**sysaddin11_l**
133	**sysaddin14_l**	134	**sysdatainvalid_l**	135	**sysaddin10_l**	136	**sysaddin12_l**
137	**sysaddin4_l**	138	**sysaddin5_l**	139	**sysaddin2_l**	140	**sysaddin6_l**
141	**sysaddout11_l**	142	**sysaddin0_l**	143	**sysaddout7_l**	144	**sysaddout9_l**
145	**sysaddout8_l**	146	**sysaddout5_l**	147	**sysaddout4_l**	148	**sysaddout3_l**
149	**sysdata63_l**	150	**sysdata62_l**	151	**sysdata61_l**	152	**sysdata59_l**
153	**sysdataoutclk7_l**	154	**sysdata53_l**	155	**syscheck6_l**	156	**sysdata55_l**
157	**sysdata56_l**	158	**sysdata58_l**	159	**sysdatainclk6_l**	160	**sysdata50_l**
161	**sysdata48_l**	162	**sysdata44_l**	163	**sysdataoutclk5_l**	164	**sysdata43_l**
165	**sysdata40_l**	166	**sysdata38_l**	167	**syscheck4_l**	168	**sysdata39_l**
169	**sysdata34_l**	170	**sysdatainclk4_l**	171	**sysdata33_l**	172	**bc_config0**
173	**bc_config2**	174	**cpu_speed0**	175	**cpu_speed2**	176	**−12v_mod**
177	**vdd_2v_term**	178	**vdd_2v_term**	179	**vdd_2v_term**	180	**vdd_2v_term**
181 to 211	**Gnd**	212	**pecl_clkin_h**	213	**pecl_clkin_l**	214	**Gnd**
215	**frameclk_h**	216	**frameclk_l**	217 to 270	**Gnd**		

4.1.2 PCI Bus Connector Pinouts

Table 4–2 shows the PCI bus connector pinouts.

Table 4–2 PCI Bus Connector Pinouts (J35, J40–J42, J44, J46)

Pin	Signal	Pin	Signal	Pin	Signal	Pin	Signal
A1	**TRST#**	A2	**+12V**	A3	**TMS**	A4	**TDI**
A5	**Vdd**	A6	**INTA**	A7	**INTC**	A8	**Vdd**
A9	—	A10	**Vdd**	A11	—	A12	**Gnd**
A13	**Gnd**	A14	—	A15	**RST#**	A16	**Vdd**
A17	**GNT#**	A18	**Gnd**	A19	—	A20	**AD[30]**
A21	**+3V**	A22	**AD[28]**	A23	**AD[26]**	A24	**Gnd**
A25	**AD[24]**	A26	**IDSEL**	A27	**+3V**	A28	**AD[22]**
A29	**AD[20]**	A30	**Gnd**	A31	**AD[18]**	A32	**AD[16]**
A33	**+3V**	A34	**FRAME#**	A35	**Gnd**	A36	**TRDY#**
A37	**STOP#**	A38	**STOP#**	A39	**+3V**	A40	**SDONE**
A41	**SBO#**	A42	**Gnd**	A43	**PAR**	A44	**AD[15]**
A45	**+3V**	A46	**AD[13]**	A47	**AD[11]**	A48	**Gnd**
A49	**AD[09]**	A50	Not used	A51	Not used	A52	**C/BE#[0]**
A53	**+3V**	A54	**AD[06]**	A55	**AD[04]**	A56	**Gnd**
A57	**AD[02]**	A58	**AD[00]**	A59	**Vdd**	A60	**REQ64#**
A61	**Vdd**	A62	**Vdd**	B1	**-12V**	B2	**TCK**
B3	**Gnd**	B4	**TDO**	B5	**Vdd**	B6	**Vdd**
B7	**INTB**	B8	**INTD**	B9	**PRSNT1#**	B10	—
B11	**PRSNT2#**	B12	**Gnd**	B13	**Gnd**	B14	—
B15	**Gnd**	B16	**CLK**	B17	**Gnd**	B18	**REQ#**
B19	**Vdd**	B20	**AD[31]**	B21	**AD[29]**	B22	**Gnd**
B23	**AD[27]**	B24	**AD[25]**	B25	**+3V**	B26	**C/BE#[3]**
B27	**AD[23]**	B28	**Gnd**	B29	**AD[21]**	B30	**AD[19]**
B31	**+3V**	B32	**AD[17]**	B33	**C/BE#[2]**	B34	**Gnd**
B35	**IRDY#**	B36	**+3V**	B37	**DEVSEL#**	B38	**Gnd**
B39	**LOCK#**	B40	**PERR#**	B41	**+3V**	B42	**SERR#**
B43	**+3V**	B44	**C/BE#[1]**	B45	**AD[14]**	B46	**Gnd**
B47	**AD[12]**	B48	**AD[10]**	B49	**Gnd**	B50	Not used
B51	Not used	B52	**AD[08]**	B53	**AD[07]**	B54	**+3V**
B55	**AD[05]**	B56	**AD[03]**	B57	**Gnd**	B58	**AD[01]**
B59	**Vdd**	B60	**ACK64#**	B61	**Vdd**	B62	**Vdd**

Table 4–2 PCI Bus Connector Pinouts (J35, J40–J42, J44, J46) (Continued)

Pin	Signal	Pin	Signal	Pin	Signal	Pin	Signal
A63	**Gnd**	A64	**C/BE#[7]**	A65	**C/BE#[5]**	A66	**Vdd**
A67	**PAR64**	A68	**D[62]**	A69	**Gnd**	A70	**D[60]**
A71	**D[58]**	A72	**Gnd**	A73	**D[56]**	A74	**D[54]**
A75	**Vdd**	A76	**D[52]**	A77	**D[50]**	A78	**Gnd**
A79	**D[48]**	A80	**D[46]**	A81	**Gnd**	A82	**D[44]**
A83	**D[42]**	A84	**Vdd**	A85	**D[40]**	A86	**D[38]**
A87	**Gnd**	A88	**D[36]**	A89	**D[34]**	A90	**Gnd**
A91	**D[32]**	A92	—	A93	**Gnd**	A94	—
B63	—	B64	**Gnd**	B65	**C/BE#[6]**	B66	**C/BE#[4]**
B67	**Gnd**	B68	**D[63]**	B69	**D[61]**	B70	**Vdd**
B71	**D[59]**	B72	**D[57]**	B73	**Gnd**	B74	**D[55]**
B75	**D[53]**	B76	**Gnd**	B77	**D[51]**	B78	**D[49]**
B79	**Vdd**	B80	**D[47]**	B81	**D[45]**	B82	**Gnd**
B83	**D[43]**	B84	**D[41]**	B85	**Gnd**	B86	**D[39]**
B87	**D[37]**	B88	**Vdd**	B89	**D[35]**	B90	**D[33]**
B91	**Gnd**	B92	—	B93	—	B94	**Gnd**

4.1.3 ISA Expansion Bus Connector Pinouts

Table 4–3 shows the ISA expansion bus connector pinouts.

Table 4–3 ISA Expansion Bus Connector Pinouts (J47)

Pin	Signal	Pin	Signal	Pin	Signal	Pin	Signal
1	**Gnd**	2	**IOCHCK#**	3	**RSTDRV**	4	**SD7**
5	**Vdd**	6	**SD6**	7	**IRQ9**	8	**SD5**
9	**–5V**	10	**SD4**	11	**DRQ2**	12	**SD3**
13	**–12V**	14	**SD2**	15	**ZEROWS#**	16	**SD1**
17	**+12V**	18	**SD0**	19	**Gnd**	20	**IOCHRDY**
21	**SMEMW#**	22	**AEN**	23	**SMEMR#**	24	**SA19**
25	**IOW#**	26	**SA18**	27	**IOR#**	28	**SA17**
29	**DACK3#**	30	**SA16**	31	**DRQ3**	32	**SA15**
33	**DACK1#**	34	**SA14**	35	**DRQ1**	36	**SA13**
37	**REFRESH#**	38	**SA12**	39	**SYSCLK**	40	**SA11**
41	**IRQ7**	42	**SA10**	43	**IRQ6**	44	**SA9**
45	**IRQ5**	46	**SA8**	47	**IRQ4**	48	**SA7**

Table 4–3 ISA Expansion Bus Connector Pinouts (J47) (Continued)

Pin	Signal	Pin	Signal	Pin	Signal	Pin	Signal
49	**IRQ3**	50	**SA6**	51	**DACK2#**	52	**SA5**
53	**TC**	54	**SA4**	55	**BALE**	56	**SA3**
57	**Vdd**	58	**SA2**	59	**OSC**	60	**SA1**
61	**Gnd**	62	**SA0**	63	**MEMCS16#**	64	**SBHE#**
65	**IOCS16#**	66	**LA23**	67	**IRQ10**	68	**LA22**
69	**IRQ11**	70	**LA21**	71	**IRQ12**	72	**LA20**
73	**IRQ15**	74	**LA19**	75	**IRQ14**	76	**LA18**
77	**DACK0#**	78	**LA17**	79	**DRQ0**	80	**MEMR#**
81	**DACK5#**	82	**MEMW#**	83	**DRQ5**	84	**SD8**
85	**DACK6#**	86	**SD9**	87	**DRQ6**	88	**SD10**
89	**DACK7#**	90	**SD11**	91	**DRQ7**	92	**SD12**
93	**Vdd**	94	**SD13**	95	**MASTER#**	96	**SD14**
97	**Gnd**	98	**SD15**	—	—	—	—

4.1.4 IDE Drive Bus Connector Pinouts

Table 4–4 shows the IDE drive bus connector pinouts.

Table 4–4 IDE Drive Bus Connector Pinouts (J45)

Pin	Signal	Pin	Signal	Pin	Signal	Pin	Signal
1	$\overline{\textbf{DRST}}$	2	**Gnd**	3	**IDE_D7**	4	**IDE_D8**
5	**IDE_D6**	6	**IDE_D9**	7	**IDE_D5**	8	**IDE_D10**
9	**IDE_D4**	10	**IDE_D11**	11	**IDE_D3**	12	**IDE_D12**
13	**IDE_D2**	14	**IDE_D13**	15	**IDE_D1**	16	**IDE_D14**
17	**IDE_D0**	18	**IDE_D15**	19	**Gnd**	20	NC (key pin)
21	**IDE_REQ0**	22	**Gnd**	23	**IDE_IOW1#**	24	**Gnd**
25	**IOR#**	26	**Gnd**	27	**CHRDY**	28	**BALE**
29	**MACK**	30	**Gnd**	31	**IRQ**	32	**IOCS16#**
33	**ADDR1**	34	NC	35	**ADDR0**	36	**ADDR2**
37	**CS0#**	38	**CS1#**	39	**ACT#**	40	**Gnd**

4.1.5 Ultra SCSI Bus Connector Pinouts

Table 4–5 shows the Ultra SCSI bus connector pinouts.

Table 4–5 Ultra SCSI Bus Connector Pinouts (J34, J38)

Pin	Signal	Pin	Signal	Pin	Signal	Pin	Signal
1	**Gnd**	2	**Gnd**	3	**Gnd**	4	**Gnd**
5	**Gnd**	6	**Gnd**	7	**Gnd**	8	**Gnd**
9	**Gnd**	10	**Gnd**	11	**Gnd**	12	**Gnd**
13	**Gnd**	14	**Gnd**	15	**Gnd**	16	**Gnd**
17	**termpwr**	18	**termpwr**	19	NC	20	**Gnd**
21	**Gnd**	22	**Gnd**	23	**Gnd**	24	**Gnd**
25	**Gnd**	26	**Gnd**	27	**Gnd**	28	**Gnd**
29	**Gnd**	30	**Gnd**	31	**Gnd**	32	**Gnd**
33	**Gnd**	34	**Gnd**	35	**scd12**	36	**scd13**
37	**scd14**	38	**scd15**	39	**scdph**	40	**scd0**
41	**scd1**	42	**scd2**	43	**scd3**	44	**scd4**
45	**scd5**	46	**scd6**	47	**scd7**	48	**scdpl**
49	**Gnd**	50	**Gnd**	51	**termpwr**	52	**termpwr**
53	NC	54	**Gnd**	55	**atn**	56	**Gnd**
57	**bsy**	58	**ack**	59	**reset**	60	**msg**
61	**sel**	62	**cd**	63	**req**	64	**io**
65	**scd8**	66	**scd9**	67	**scd10**	68	**scd11**

4.1.6 SDRAM DIMM Connector Pinouts

Table 4–6 shows the SDRAM DIMM connector pinouts.

Table 4–6 SDRAM DIMM Connector Pinouts (J1, J5, J6, J9, J11, J13, J14, J16, J25–J32)

Pin	Signal	Pin	Signal	Pin	Signal	Pin	Signal
1	**Vdd**	2	NC	3	NC	4	NC
5	NC	6	NC	7	NC	8	**Vss**
9	**dq67**	10	**dq66**	11	**Vdd**	12	**dq65**
13	**dq64**	14	**Vss**	15	**dq63**	16	**dq62**
17	NC	18	**dq61**	19	**dq60**	20	**Vdd**
21	NC	22	NC	23	**Vss**	24	NC
25	NC	26	**Vdd**	27	**dq51**	28	**dq50**

Table 4–6 SDRAM DIMM Connector Pinouts (J1, J5, J6, J9, J11, J13, J14, J16, J25–J32) (Continued)

Pin	Signal	Pin	Signal	Pin	Signal	Pin	Signal
29	**Vss**	30	**dq49**	31	**dq48**	32	**Vdd**
33	**dq43**	34	**dq42**	35	**Vss**	36	**dq41**
37	**dq40**	38	**Vdd**	39	**a4**	40	**a5**
41	**Vss**	42	**a8**	43	**a9**	44	**Vdd**
45	NC	46	**cke0**	47	**Vss**	48	**cas#**
49	NC	50	**Vdd**	51	**Vss**	52	**ras#**
53	**Vss**	54	**cs2#**	55	**a11**	56	**Vdd**
57	**a0**	58	**a1**	59	**Vss**	60	**dq35**
61	**dq34**	62	**Vdd**	63	**dq33**	64	**dq32**
65	**Vss**	66	**dq27**	67	**dq26**	68	**Vdd**
69	**dq25**	70	**dq24**	71	**Vss**	72	**dq19**
73	**dq18**	74	**Vdd**	75	**dq17**	76	**dq16**
77	**Vss**	78	NC	79	NC	80	**Vdd**
81	**dq15**	82	**dq14**	83	**Vss**	84	**dq13**
85	**dq12**	86	**Vdd**	87	**dq7**	88	**dq6**
89	**Vss**	90	**dq5**	91	**dq4**	92	**Vdd**
93	NC	94	NC	95	NC	96	NC
97	NC	98	**scl**	99	NC	100	**Vss**
101	NC	102	NC	103	**Vss**	104	**rege**
105	**rfu**	106	**rfu**	107	NC	108	**dq71**
109	**dq70**	110	**Vss**	111	**dq69**	112	**dq68**
113	**Vdd**	114	NC	115	**Vss**	116	NC
117	**dq59**	118	**dq58**	119	**Vss**	120	**dq57**
121	**dq56**	122	**Vdd**	123	**dq55**	124	**dq54**
125	**Vss**	126	**dq53**	127	**dq52**	128	**Vdd**
129	**dq47**	130	**dq46**	131	**Vss**	132	**dq45**
133	**dq44**	134	**Vdd**	135	**dq39**	136	**dq38**
137	**Vss**	138	**dq37**	139	**dq36**	140	**Vdd**
141	**a6**	142	**a7**	143	**Vss**	144	**bs0**
145	NC	146	**Vdd**	147	**dqm**	148	**we#**
149	**Vss**	150	NC	151	**clk0**	152	**Vdd**
153	NC	154	**cs0#**	155	**Vss**	156	**ba1**
157	**a10/ap**	158	**Vdd**	159	**a2**	160	**a3**
161	**Vss**	162	**dq31**	163	**dq30**	164	**Vdd**

Table 4–6 SDRAM DIMM Connector Pinouts (J1, J5, J6, J9, J11, J13, J14, J16, J25–J32) (Continued)

Pin	Signal	Pin	Signal	Pin	Signal	Pin	Signal
165	**dq29**	166	**dq28**	167	**Vss**	168	**dq23**
169	**dq22**	170	**Vdd**	171	**dq21**	172	**dq20**
173	**Vss**	174	NC	175	NC	176	**Vdd**
177	NC	178	**Vss**	179	**Vss**	180	NC
181	NC	182	**Vdd**	183	**dq11**	184	**dq10**
185	**Vss**	186	**dq9**	187	**dq8**	188	**Vdd**
189	**dq3**	190	**dq2**	191	**Vss**	192	**dq1**
193	**dq0**	194	**sda**	195	**sa0**	196	**sa1**
197	**sa2**	198	**Vdd**	199	NC	200	NC

4.1.7 Diskette (Floppy) Drive Bus Connector Pinouts

Table 4–7 shows the diskette (floppy) drive bus connector pinouts.

Table 4–7 Diskette (Floppy) Drive Bus Connector Pinouts (J43)

Pin	Signal	Pin	Signal	Pin	Signal	Pin	Signal
1	**Gnd**	2	**DRVDEN0**	3	**Gnd**	4	NC
5	**Gnd**	6	**DRVDEN1**	7	**Gnd**	8	**INDEX**
9	**Gnd**	10	**MTR0**	11	**Gnd**	12	**DS1**
13	**Gnd**	14	**DS0**	15	**Gnd**	16	**MTR1**
17	**Gnd**	18	**DIR**	19	**Gnd**	20	**STEP**
21	**Gnd**	22	**WDATA**	23	**Gnd**	24	**WGATE**
25	**Gnd**	26	**TRK0**	27	**Gnd**	28	**WRTPRT**
29	**Gnd**	30	**RDATA**	31	**Gnd**	32	**HDSEL**
33	**Gnd**	34	**DSKCHG**	—	—	—	—

4.1.8 Parallel Bus Connector Pinouts

Table 4–8 shows the parallel bus connector pinouts.

Table 4–8 Parallel Bus Connector Pinouts (J17)

Pin	Signal	Pin	Signal	Pin	Signal	Pin	Signal
1	**PSTB**	2	**PD0**	3	**PD1**	4	**PD2**
5	**PD3**	6	**PD4**	7	**PD5**	8	**PD6**
9	**PD7**	10	**PACK**	11	**PBUSY**	12	**PE**

Table 4–8 Parallel Bus Connector Pinouts (J17) (Continued)

Pin	Signal	Pin	Signal	Pin	Signal	Pin	Signal
13	**PSLCT**	14	**PAFD**	15	**PAR_ERROR**	16	**PINIT**
17	**PSLIN**	18	**Gnd**	19	**Gnd**	20	**Gnd**
21	**Gnd**	22	**Gnd**	23	**Gnd**	24	**Gnd**
25	**Gnd**	—	—	—	—	—	—

4.1.9 COM1/COM2 Serial Line Connector Pinouts

Table 4–9 shows the COM1/COM2 serial line connector pinouts.

Table 4–9 COM1/COM2 Serial Line Connector Pinouts (J19)

COM1 Pin (Top)	COM1 Signal	COM2 Pin (Bottom)	COM2 Signal
1	**DCD1**	10	**DCD2**
2	**SIN1**	11	**SIN2**
3	**SOUT1**	12	**SOUT2**
4	**DTR1**	13	**DTR2**
5	**Gnd**	14	**Gnd**
6	**DSR1**	15	**DSR2**
7	**RTS1**	16	**RTS2**
8	**CTS1**	17	**CTS2**
9	**RI1**	18	**RI2**

4.1.10 Keyboard/Mouse Connector Pinouts

Table 4–10 shows the keyboard/mouse connector pinouts.

Table 4–10 Keyboard/Mouse Connector Pinouts (J21)

Keyboard Pin (Top)	Keyboard Signal	Mouse Pin (Bottom)	Mouse Signal
1	**KBDATA**	7	**MSDATA**
2	NC	8	NC
3	**Gnd**	9	**Gnd**
4	**Vdd**	10	**Vdd**
5	**KBCLK**	11	**MSCLK**
6	NC	12	NC

4.1.11 +3-V Power Connector Pinouts

Table 4–11 shows the +3-V power connector pinouts.

Table 4–11 +3-V Power Connector Pinouts (J3)

Pin	Voltage	Pin	Voltage	Pin	Voltage	Pin	Voltage
1	**vdd_3v**	2	**Gnd**	3	**vdd_3v**	4	**Gnd**
5	**vdd_3v**	6	**Gnd**	7	**vdd_3v**	8	**Gnd**
9	**vdd_3v**	10	**Gnd**	11	**vdd_3v**	12	**Gnd**
13	**Gnd**	14	**vdd_3v**	15	**Gnd**	16	**vdd_3v**
17	**Gnd**	18	**vdd_3v**	19	**Gnd**	20	**vdd_3v**
21	+12 V dc	22	**ps_on**	23	**pok**	24	**5vsb**

4.1.12 +5-V Power Connector Pinouts

Table 4–12 shows the +5-V power connector pinouts.

Table 4–12 +5-V Power Connector Pinouts (J33)

Pin	Voltage	Pin	Voltage	Pin	Voltage	Pin	Voltage
1	**vdd_5v**	2	**Gnd**	3	**vdd_5v**	4	**Gnd**
5	**vdd_5v**	6	**Gnd**	7	**vdd_5v**	8	**Gnd**
9	**Gnd**	10	**vdd_5v**	11	**Gnd**	12	**vdd_5v**
13	**Gnd**	14	**vdd_5v**	15	-12 V dc	16	-5 V dc

4.1.13 Fan Box Power Connector Pinouts

Table 4–13 shows the fan box power connector pinouts.

Table 4–13 Fan Box Power Connector Pinouts (J2, J15, J22, J24)

Pin	Voltage	Pin	Voltage
1	**Gnd**	2	+12 V dc

4.1.14 Speaker Connector Pinouts

Table 4–14 shows the speaker connector pinouts.

Table 4–14 Speaker Connector Pinouts (J39)

Pin	Signal	Description
1	**spkr**	Speaker input
2	**vdd_5v**	—
3	**Gnd**	—
4	**vdd_5v**	—

4.1.15 Halt Button Connector Pinouts

Table 4–15 shows the halt button connector pinouts.

Table 4–15 Halt Button Connector Pinouts (J12)

Pin	Signal	Description
1	**halt_button**	Halt system (for Tru64 UNIX only)
2	**vdd_5v**	—

4.1.16 Reset Button Connector Pinouts

Table 4–16 shows the reset button connector pinouts.

Table 4–16 Reset Button Connector Pinouts (J8)

Pin	Signal	Description
1	**reset_button**	Reset system
2	**vdd_5v**	—

4.1.17 System Power Button Connector Pinouts

Table 4–17 shows the system power button connector pinouts.

Table 4–17 System Power Button Connector Pinouts (J7)

Pin	Signal	Description
1	**5vsb**	System power on/off
2	**Gnd**	—

4.1.18 Ultra SCSI Hard Drive LED Connector Pinouts

Table 4–18 shows the ultra SCSI hard drive LED connector pinouts.

Table 4–18 Ultra SCSI Hard Drive LED Connector Pinouts (J10)

Pin	Signal	Description
1	**scsi_hd_act**	Hard drive active
2	**vdd_5v**	—

4.1.19 Power LED Connector Pinouts

Table 4–19 shows the power LED connector pinouts.

Table 4–19 Power LED Connector Pinouts (J36)

Pin	Signal	Description
1	**power_led**	Power LED input
2	**Gnd**	—
3	NC	—
4	NC	—
5	NC	—

4.2 Daughtercard Connectors

Tables 4–20 through 4–23 show the daughtercard connector pinouts. See Figure 2–2 for connector locations.

4.2.1 Microprocessor Fan Power Connector Pinouts

Table 4–20 shows the microprocessor fan power connector pinouts.

Table 4–20 Microprocessor Fan Power Connector Pinouts (J1)

Pin	Signal	Description
1,6	+12 V dc	—
2,5	**Gnd**	—
3,4	**fan_conn_l**	Fan connected

4.2.2 SROM Test Data Input Connector Pinouts

Table 4–21 shows the SROM test data input connector pinouts.

Table 4–21 SROM Test Data Input Connector Pinouts (J2)

Pin	Signal	Description
1	NC	—
2	**srom_clk_l**	Clock out
3	**Gnd**	—
4	NC	—
5	**test_srom_d_l**	SROM serial data in
6	NC	—

4.2.3 AlphaPC 264DP Daughtercard Connector Pinouts

Table 4–22 shows the AlphaPC 264DP daughtercard connector pinouts.

Table 4–22 Daughtercard Connector Pinouts (J3)

Pin	Signal	Pin	Signal	Pin	Signal	Pin	Signal
1	**vdd_3v**	2	**vdd_3v**	3	**vdd_3v**	4	**vdd_3v**
5	**vdd_3v**	6	**vdd_3v**	7	**vdd_3v**	8	**vdd_3v**
9	**sysdata2_l**	10	**sysdata0_l**	11	**sysdata5_l**	12	**sysdata3_l**
13	**syscheck0_l**	14	**sysdata9_l**	15	**sysdata10_l**	16	**sysdata13_l**
17	**sysdata14_l**	18	**sysdataoutclk1_l**	19	**sysdata16_l**	20	**syscheck20_l**
21	**sysdata17_l**	22	**sysdata18_l**	23	**sysdataoutclk2_l**	24	**sysdata21_l**
25	**sda**	26	**Gnd**	27	**Gnd**	28	**cpu_slot**
29	**sysdata26_l**	30	**sysdata25_l**	31	**sysdatainclk3_l**	32	**sysdata28_l**
33	**sysdata30_l**	34	**syscheck3_l**	35	**clkfwdreset_h**	36	**irq_0_h**
37	**irq_2_h**	38	**irq_4_h**	39	**2v_pwrgood_h**	40	**tsu_speed0**
41	**tsu_speed1**	42	**clk_rdy_h**	43	**tsu_speed2**	44	**sysfillvalid_l**
45	**sysaddin13_l**	46	**sysaddin8_l**	47	**sysaddin9_l**	48	**sysaddinclk_l**
49	**sysaddout13_l**	50	**sysaddin7_l**	51	**sysaddin1_l**	52	**sysaddin3_l**
53	**sysaddout14_l**	54	**sysaddout12_l**	55	**sysaddout10_l**	56	**sysaddoutclk_l**
57	**sysaddout6_l**	58	**sysaddout0_l**	59	**sysaddout2_l**	60	**syscheck7_l**
61	**sysaddout1_l**	62	**sysdata60_l**	63	**sysdatainclk7_l**	64	**sysdata52_l**
65	**sysdata54_l**	66	**sysdata57_l**	67	**sysdataoutclk6_l**	68	**sysdata51_l**
69	**sysdata49_l**	70	**syscheck5_l**	71	**sysdata46_l**	72	**sysdata47_l**
73	**sysdata45_l**	74	**sysdata42_l**	75	**sysdatainclk5_l**	76	**sysdata41_l**

Table 4–22 Daughtercard Connector Pinouts (J3) (Continued)

Pin	Signal	Pin	Signal	Pin	Signal	Pin	Signal
77	**sysdataoutclk4_l**	78	**sysdata37_l**	79	**sysdata36_l**	80	**sysdata35_l**
81	**sysdata32_l**	82	**Gnd**	83	**bc_config1**	84	**bc_config3**
85	**cpu_speed1**	86	**+12v_mod**	87	**vdd_2v_term**	88	**vdd_2v_term**
89	**vdd_2v_term**	90	**vdd_2v_term**	91	**vdd_3v**	92	**vdd_3v**
93	**vdd_3v**	94	**vdd_3v**	95	**vdd_3v**	96	**vdd_3v**
97	**vdd_3v**	98	**vdd_3v**	99	**sysdatainclk0_l**	100	**sysdata6_l**
101	**sysdata1_l**	102	**sysdataoutclk0_l**	103	**sysdata4_l**	104	**sysdata7_l**
105	**sysdata8_l**	106	**sysdatainclk1_l**	107	**sysdata11_l**	108	**sysdata12_l**
109	**sysdata15_l**	110	**sysdata19_l**	111	**sysdatainclk2_l**	112	**sysdata20_l**
113	**sysdata22_l**	114	**sysdata23_l**	115	**pllbypass_h**	116	**srom_en_l**
117	**sclk**	118	**syscheck2_l**	119	**sysdata24_l**	120	**sysdata27_l**
121	**sysdataoutclk3_l**	122	**sysdata29_l**	123	**sysdata31_l**	124	**sysdataoutvalid_l**
125	**Gnd**	126	**Gnd**	127	**fan_ok_l**	128	**irq_1_h**
129	**irq_3_h**	130	**irq_5_h**	131	**mod_reset_l**	132	**sysaddin11_l**
133	**sysaddin14_l**	134	**sysdatainvalid_l**	135	**sysaddin10_l**	136	**sysaddin12_l**
137	**sysaddin4_l**	138	**sysaddin5_l**	139	**sysaddin2_l**	140	**sysaddin6_l**
141	**sysaddout11_l**	142	**sysaddin0_l**	143	**sysaddout7_l**	144	**sysaddout9_l**
145	**sysaddout8_l**	146	**sysaddout5_l**	147	**sysaddout4_l**	148	**sysaddout3_l**
149	**sysdata63_l**	150	**sysdata62_l**	151	**sysdata61_l**	152	**sysdata59_l**
153	**sysdataoutclk7_l**	154	**sysdata53_l**	155	**syscheck6_l**	156	**sysdata55_l**
157	**sysdata56_l**	158	**sysdata58_l**	159	**sysdatainclk6_l**	160	**sysdata50_l**
161	**sysdata48_l**	162	**sysdata44_l**	163	**sysdataoutclk5_l**	164	**sysdata43_l**
165	**sysdata40_l**	166	**sysdata38_l**	167	**syscheck4_l**	168	**sysdata39_l**
169	**sysdata34_l**	170	**sysdatainclk4_l**	171	**sysdata33_l**	172	**bc_config0**
173	**bc_config2**	174	**cpu_speed0**	175	**cpu_speed2**	176	**−12v_mod**
177	**vdd_2v_term**	178	**vdd_2v_term**	179	**vdd_2v_term**	180	**vdd_2v_term**
181 to 211	**Gnd**	212	**pecl_clkin_h**	213	**pecl_clkin_l**	214	**Gnd**
215	**frameclk_h**	216	**frameclk_l**	217 to 270	**Gnd**		

4.2.4 AlphaPC 264DP Daughtercard Input Power Connector Pinouts

Table 4–23 shows the input power connector pinouts.

Table 4–23 Input Power Connector Pinouts (J4)

Pin	Voltage	Pin	Voltage	Pin	Voltage	Pin	Voltage
1	+5 V dc	2	**Gnd**	3	+5 V dc	4	**Gnd**
5	+5 V dc	6	**Gnd**	7	+5 V dc	8	**Gnd**
9	+5 V dc	10	**Gnd**	11	+5 V dc	12	**Gnd**
13	+5 V dc	14	**Gnd**	15	+5 V dc	16	**Gnd**
17	+5 V dc	18	**Gnd**				

5
Memory Configuration

5.1 Memory Subsystem

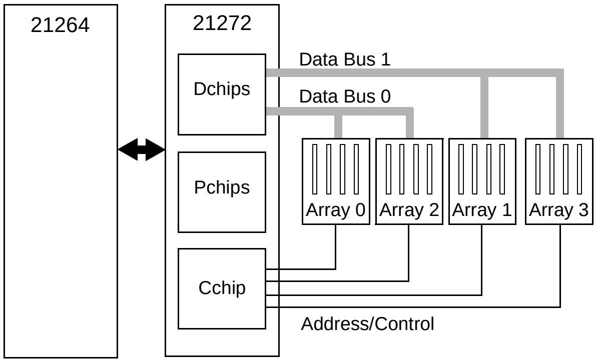
The DRAM memory subsystem on the AlphaPC 264DP consists of sixteen 200-pin buffered DIMM slots, which are organized as four arrays of memory. The 21272 chipset supports two 256-bit memory buses (288-bit including ECC) with two arrays on each bus (see Figure 5–1).

The 72-bit, 100-MHz DIMMs consist of 64 bits of data and 8 bits of ECC, and can be 32MB, 64MB, 128MB, or 256MB. The minimum configuration (one array populated with four 32MB DIMMs) is 128MB. The maximum configuration (four arrays each populated with four 256MB DIMMs) is 4GB.

The memory cycle time is 83 MHz, identical to the 21272 chipset cycle time.

Note: Although the memory cycle time is 83 MHz, qualified 100-MHz DIMMs are required.

Figure 5–1 AlphaPC 264DP Memory Subsystem



5.2 Configuring SDRAM Memory

For the memory system in the AlphaPC 264DP, one to four arrays may be used, following the configuration rules.

Configuration Rules

- Each array must be fully populated with DIMMs of the same size and type.
- Array 0 must be populated.
- Additional arrays can be populated in any order.

  For a memory subsystem with two arrays, placing the second array on bus 1 (array 1 or array 3) is recommended.

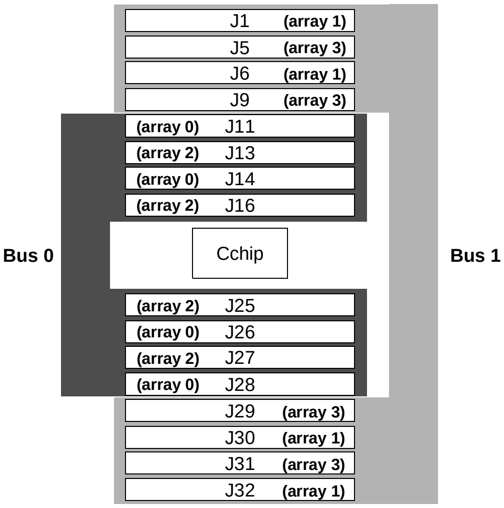
Arrays

The arrays are made up of the following connectors:

- **Array 0**: J11, J14, J26, J28
- **Array 1**: J1, J6, J30, J32
- **Array 2**: J13, J16, J25, J27
- **Array 3**: J5, J9, J29, J31

Figure 5–2 shows the relationship of the connectors/arrays. Refer to Figure 2–1 for DIMM connector locations on the mainboard.

Figure 5–2 AlphaPC 264DP DIMM Connectors



Configuring SDRAM Memory

Possible Configurations

Memory sizes from 128MB to 4GB are supported.

Although not an exhaustive list, Table 5–1 lists some of the SDRAM memory configurations available. Any combinations of DIMMs that meet the configuration rules are supported by the 21272 chipset.

For a list of vendors who supply components and accessories for the AlphaPC 264DP, see Appendix A.

Table 5–1 AlphaPC 264DP SDRAM Memory Configurations

Total Memory	Array 0[1]	Array 1[1]	Array 2[1]	Array 3[1]
128MB	32MB	—	—	—
256MB	32MB	32MB	—	—
	64MB	—	—	—
512MB	128MB	—	—	—
	64MB	64MB	—	—
768MB	128MB	64MB	—	—
	64MB	64MB	64MB	—
1GB	256MB	—	—	—
	128MB	128MB	—	—
	64MB	64MB	64MB	64MB
1.5GB	256MB	128MB	—	—
	128MB	128MB	64MB	64MB
2GB	256MB	256MB	—	—
2.5GB	256MB	256MB	128MB	—
3GB	256MB	256MB	128MB	128MB
3.5GB	256MB	256MB	256MB	128MB
4GB	256MB	256MB	256MB	256MB

[1] Each array has 4 DIMMs.

6 Firmware Introduction

This chapter introduces the operating system interfaces (firmware) that are stored in the AlphaPC 264DP flash ROM:

- AlphaBIOS
- Alpha SRM console

It also describes two additional utilities that are used with the AlphaPC 264DP:

- The Fail-Safe Booter (FSB) utility
- The Loadable Firmware Update (LFU) utility

6.1 Firmware Overview

There are two interfaces that are included in the AlphaPC 264DP firmware, residing in the flash ROM on the AlphaPC 264DP mainboard. They are defined as follows:

- **AlphaBIOS** — The AlphaBIOS firmware initializes the system and allows you to configure the system and install and boot the Windows NT operating system.
- **Alpha SRM console** — The Alpha SRM console firmware initializes the system and loads the AlphaBIOS firmware. It also allows you to configure the system and install and boot the Tru64 UNIX operating system.

6.1.1 Power-Up

The Alpha SRM console is loaded at power-up to perform a few functions. The Alpha SRM console then loads AlphaBIOS and passes control to it.

6.1.2 Switching From One Firmware to the Other

At times, you may find it necessary to switch from one firmware to the other. The following sections describe the necessary steps.

From AlphaBIOS to Alpha SRM Console

To boot the Alpha SRM console while in AlphaBIOS, perform the following steps. (For more information, refer to Chapter 7.)

1. At the AlphaBIOS boot screen, press F2 to reach the AlphaBIOS setup screen.
2. From the setup screen, use the Tab key to select CMOS Setup...
3. At the CMOS setup screen, press F6 to move to the Advanced CMOS setup screen.
4. At the Advanced CMOS setup screen, use the Tab key to move to Console Selection. Then use the down arrow key to toggle the console selection from *Windows NT Console (AlphaBIOS)* to *Tru64 UNIX (SRM Console).*
5. Press F10 to save the changes and return to the CMOS setup screen.
6. Press F10 to save the changes and return to the AlphaBIOS setup screen.
7. Initialize the system by pressing the reset button or by power cycling the system.

From Alpha SRM Console to AlphaBIOS

To boot AlphaBIOS while in the Alpha SRM console, perform the following steps. (For more information, refer to Chapter 8.)

1. At the Alpha SRM console prompt, enter the command **set os_type nt**.
2. Initialize the system by pressing the reset button or entering the **init** command.

6.2 Preparing Diskettes

The required firmware for your system is preloaded onto the flash ROM. Copies of the firmware files may be included on your distribution CD, in case you need to refresh the firmware. If they are not included, you can download them from the Alpha OEM World Wide Web Internet site at **http://www.digital.com/alphaoem**. Click on **Technical Information**, then click on **Alpha Drivers and Firmware**.

The utilities that are used to reload or update the firmware expect to find the files on a diskette, so you need to prepare a diskette for each utility with the correct files from the CD or the Web.

For FSB:

- Copy the file PC264SRM.ROM onto a diskette, renaming it DP264SRM.ROM.

For Updating Firmware:

- Copy the files PC264FW.TXT, PC264NT.ROM, and PC264SRM.ROM onto a diskette.

For Installing Windows NT:

- Copy the necessary files, as directed by installation instructions, onto a diskette (see Section 7.3.1).

6.3 Fail-Safe Booter Utility

The FSB provides an emergency recovery mechanism when the firmware image contained in flash memory has been corrupted. You can run the FSB and boot another image from a diskette that is capable of reprogramming the flash ROM.

You can start the FSB in one of two ways:

- If the firmware image is unavailable when the system is powered on or reset, the FSB runs automatically. When the FSB runs, the system emits a series of beeps through the speaker as beep code 1-2-3; that is, one beep and a pause, followed by two beeps and a pause, followed by three beeps. After the diskette activity light flashes, insert the diskette you created for the FSB. Then, reset the system to restart the FSB. The FSB loads and runs the Alpha SRM console from this diskette. Go to Section 6.5.1 and follow the procedures for updating your firmware.
- You can also start the FSB manually, as follows:
  1. Power off your system.
  2. Set switch 1 (FSB) of SW2 on the mainboard to on (see Figure 3–1).
  3. Insert the FSB diskette you created into the diskette drive.
  4. Power on your system, which brings up the Alpha SRM console.
  5. Proceed to Section 6.5.1 and follow the procedures for updating your firmware.

6.4 Updating Firmware From AlphaBIOS

At the the AlphaBIOS Setup screen, load the firmware update diskette you created, then use the Tab key to select the AlphaBIOS Upgrade option and press Enter (see Section 7.2.2). The AlphaBIOS automatically finds the firmware update files and updates the appropriate firmware.

6.5 Updating Firmware From Alpha SRM Console with LFU

To update the firmware from the Alpha SRM console, start the Loadable Firmware Update (LFU) utility, which then allows you to update the Alpha SRM console firmware, the AlphaBIOS firmware, and option firmware.

6.5.1 Starting LFU

At the Alpha SRM console prompt, issue the **lfu** command (see Section 8.11).

The LFU, upon being started, searches for the update firmware. You need to perform the following steps to update the AlphaBIOS and the Alpha SRM console.

1. Load the diskette you created for the LFU.
2. Enter the device name **dva0** when prompted for the location of the update files.
3. Enter the filename PC264FW.TXT when prompted. Note that the LFU has already checked the contents of the diskette and should provide PC264FW.TXT as the default.

PC264FW.TXT specifies which firmware is to be updated and passes the names of the files that contain updated firmware. Figure 6–1 shows an example of running LFU and updating firmware.

Figure 6–1 Example of Running LFU

```
P00>>>lfu
Checking dka400.4.0.7.1 for the option firmware files. . .
Checking dva0 for the option firmware files. . .

Option firmware files were not found on CD or floppy.
If you want to load the options firmware,
please enter the device on which the files are located(ewa0),
or just press <return> to proceed with a standard console update: dva0
Please enter the name of the options firmware files list, or
Press <return> to use the default filename (pc264fw.txt) : pc264fw.txt
Copying PC264FW.TXT from dva0. . .
Copying PC264NT.ROM from dva0. . .
Copying PC264SRM.ROM from dva0. . .
```

6.5.2 LFU Commands

Table 6–1 shows the commands available in the LFU.

Table 6–1 LFU Commands

Command	Description
display	Displays the system’s configuration table
exit	Exits LFU
list	Lists the device, revision, firmware name, and update revision
readme	Displays important release information
update	Replaces the current firmware with the loadable data image
verify	Compares the loadable and hardware images
? or help	Displays this command table

display

The **display** command shows the physical configuration of the system. **Display** is equivalent to issuing the SRM console command **show configuration**. It shows you the slot for each module, so **display** can help you identify the location of a device.

exit

The **exit** command terminates the LFU program, causes system initialization and testing, and returns the system to the console from which LFU was called.

help

The **help** (or **?**) command displays the LFU command list, shown below.

```
Function    Description

Display     Displays the system’s configuration table.
Exit        Done exit LFU (reset).
List        Lists the device, revision, firmware name, and update revision.
Readme      Lists important release information
Update      Replaces current firmware with loadable data image.
Verify      Compares loadable and hardware image.
? or Help   Scrolls this function table.
```

list

The **list** command displays the inventory of update firmware on the CD-ROM, floppy, or network. Only the devices listed at your terminal are supported for firmware updates.

The **list** command shows three pieces of information for each device:

- Current Revision – The revision of the device's current firmware
- Filename – The name of the file used to update that firmware
- Update Revision – The revision of the firmware update image

readme

The **readme** command lists release notes for the LFU program.

update

The **update** command writes new firmware to the module. Then LFU automatically verifies the update by reading the new firmware image from the module into memory and comparing it with the source image.

To update more than one device, you may use a wildcard but not a list. For example, *update k** updates all devices with names beginning with k, and *update ** updates all devices.

When you do not specify a device name, LFU tries to update all devices; it lists the selected devices to update and prompts before devices are updated. The default is no.

The *-all* option eliminates the update confirmation requests, enabling the update to proceed without operator intervention.

Caution: Never abort an update operation. Aborting corrupts the firmware on the module.

verify

The **verify** command reads the firmware from the module into memory and compares it with the update firmware. If a module was verified when it was updated, but later failed tests, use **verify** to determine whether the firmware has become corrupted.

7

AlphaBIOS

The AlphaBIOS firmware initializes the system and enables you to install and boot the Windows NT operating system. This firmware resides in the flash ROM on the AlphaPC 264DP mainboard. The mainboard ships with AlphaBIOS firmware and online documentation that describes how to configure the firmware for Windows NT.

This chapter contains a subset of the AlphaBIOS help file on the distribution medium. You can access the complete documentation online when your system is up and running, or from another Windows system.

The online documentation describes configuration information for Alpha systems running AlphaBIOS. The online documentation resides on the distribution medium. The documentation is available as a Windows Help file compatible with Windows NT 4.0 and Windows 95.

7.1 AlphaBIOS Conventions

AlphaBIOS uses universally accepted keys and key combinations for navigating the interface and selecting items. If you are familiar with MS-DOS or Microsoft Windows keyboard conventions, navigating AlphaBIOS is simple. Table 7–1 shows the keys to use for navigating and selecting items in AlphaBIOS.

Table 7–1 Action Keys

Press...	To...
TAB	Move highlight forward between fields of a dialog
SHIFT + TAB	Move highlight backwards between fields of a dialog
↑ or ↓	Move highlight within a menu, or cycle through available field values in a dialog window
ALT + ↓	Drop down a menu of choices from a drop-down listbox. A drop-down listbox can be recognized by the symbol ⇓
HOME	Move to the beginning of a text entry field
END	Move to the end of a text entry field
← →	Move to the left or right in a text entry field
ESC	Discard changes and back up to previous screen

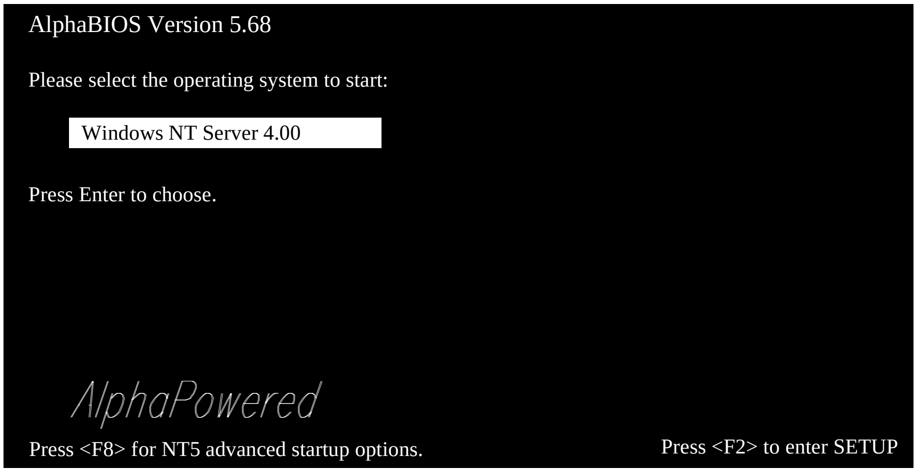
Two levels of keyboard help are available:

- Press **F1** once to display explanations of the keystrokes available for the *currently displayed* part of AlphaBIOS.
- Press **F1** twice to display explanations of the keystrokes available for navigating *throughout* AlphaBIOS.

7.2 AlphaBIOS Setup Program

When you power up or reset your system, the boot screen with the system logo is displayed. Pressing **F2** at this screen starts the setup program. Figure 7–1 shows an example of an AlphaBIOS Boot Screen with the “Press <F2> to enter SETUP” message at the bottom. Figure 7–2 shows the AlphaBIOS Setup Screen.

Figure 7–1 AlphaBIOS Boot Screen



AlphaBIOS Version 5.68

Please select the operating system to start:

Windows NT Server 4.00

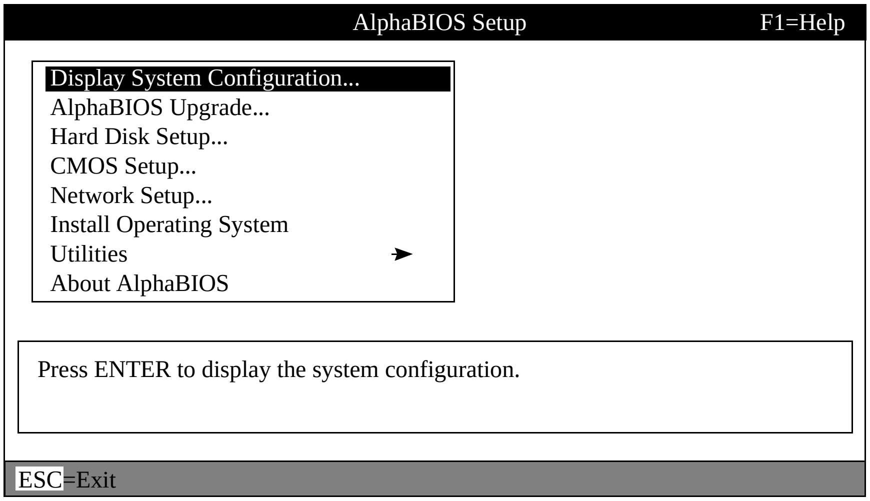
Press Enter to choose.

AlphaPowered

Press <F8> for NT5 advanced startup options.

Press <F2> to enter SETUP

Figure 7–2 AlphaBIOS Setup Screen



AlphaBIOS Setup F1=Help

Display System Configuration...
AlphaBIOS Upgrade...
Hard Disk Setup...
CMOS Setup...
Network Setup...
Install Operating System
Utilities ➤
About AlphaBIOS

Press ENTER to display the system configuration.

ESC=Exit

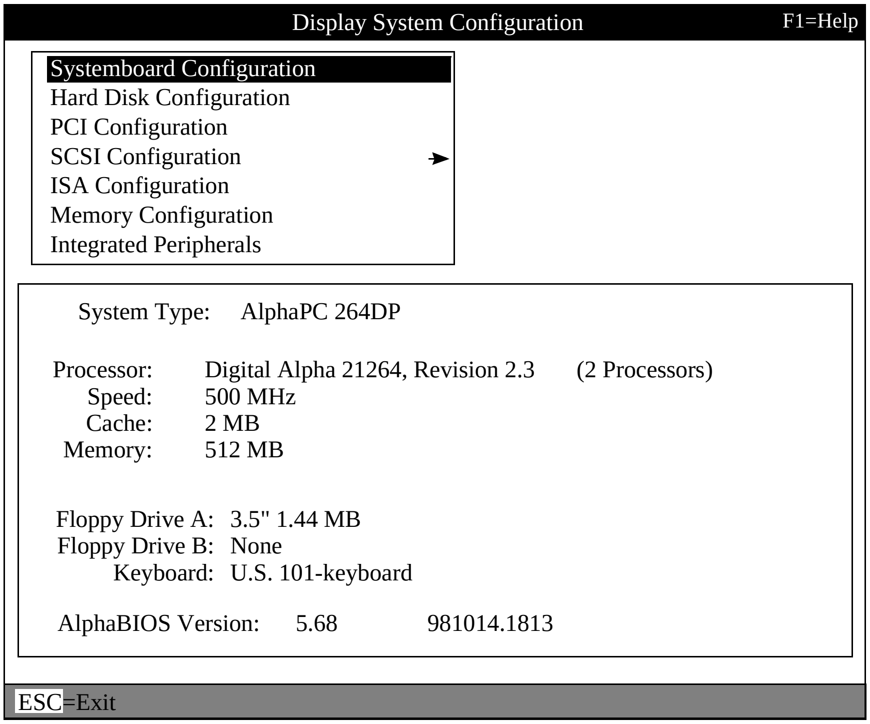
7.2.1 Displaying Your System Configuration

You can use AlphaBIOS to display your system configuration in a clear, easy-to-read format, organized by categories: system board, memory, hard disk, PCI, SCSI, and peripherals.

To display your system configuration:

1. Start AlphaBIOS Setup, select Display System Configuration, and press **Enter**.
2. In the Display System Configuration screen, use the arrow keys to select the configuration category that you want to see, such as the examples shown in Figures 7–3 through 7–5.

Figure 7–3 Systemboard Configuration Screen



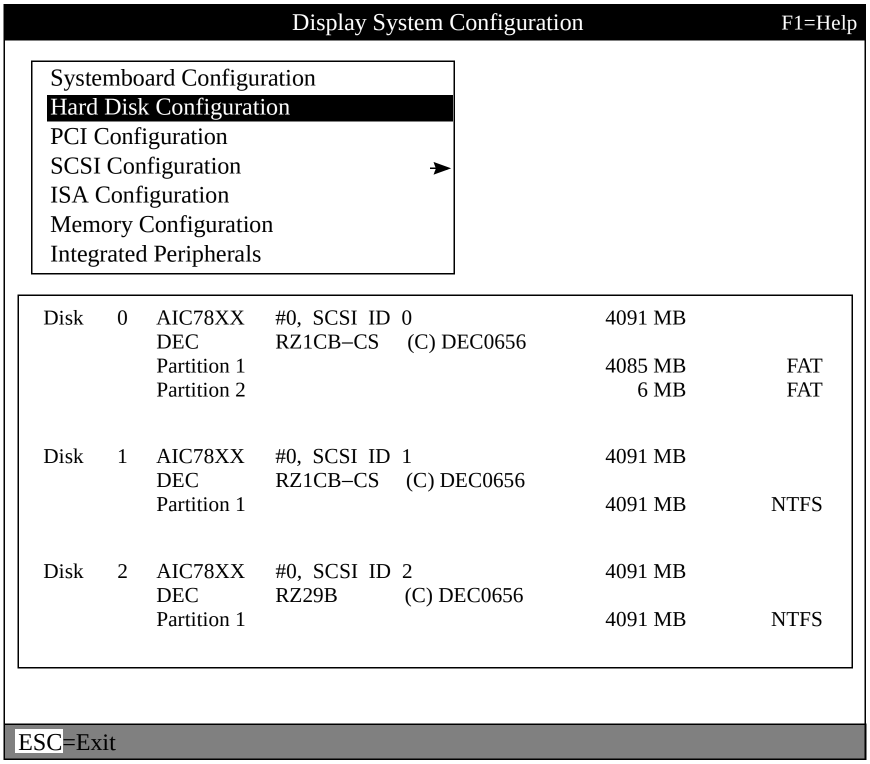
```
    System Type:     AlphaPC 264DP

  Processor:       Digital Alpha 21264, Revision 2.3      (2 Processors)
      Speed:       500 MHz
      Cache:       2 MB
     Memory:       512 MB


  Floppy Drive A:  3.5" 1.44 MB
  Floppy Drive B:  None
        Keyboard:  U.S. 101-keyboard

  AlphaBIOS Version:     5.68            981014.1813
```

Figure 7–4 Hard Disk Configuration Screen



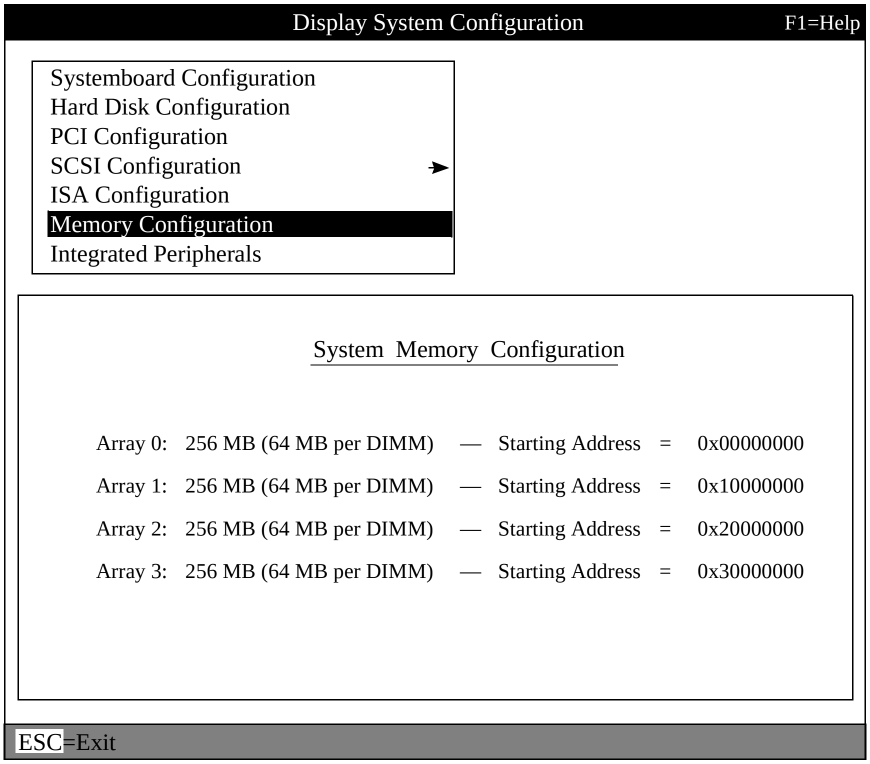
```
Disk   0   AIC78XX    #0,  SCSI  ID  0                 4091 MB
           DEC        RZ1CB–CS     (C) DEC0656
           Partition 1                                 4085 MB        FAT
           Partition 2                                    6 MB        FAT


Disk   1   AIC78XX    #0,  SCSI  ID  1                 4091 MB
           DEC        RZ1CB–CS     (C) DEC0656
           Partition 1                                 4091 MB       NTFS


Disk   2   AIC78XX    #0,  SCSI  ID  2                 4091 MB
           DEC        RZ29B        (C) DEC0656
           Partition 1                                 4091 MB       NTFS
```

Note: This screen is for information only; it cannot be edited. To make changes to the hard-disk setup, use the Hard Disk Setup screen (see Section 7.2.3).

Figure 7–5 Memory Configuration Screen



```
                 System  Memory  Configuration

Array 0:  256 MB (64 MB per DIMM)   —  Starting Address  =  0x00000000
Array 1:  256 MB (64 MB per DIMM)   —  Starting Address  =  0x10000000
Array 2:  256 MB (64 MB per DIMM)   —  Starting Address  =  0x20000000
Array 3:  256 MB (64 MB per DIMM)   —  Starting Address  =  0x30000000
```

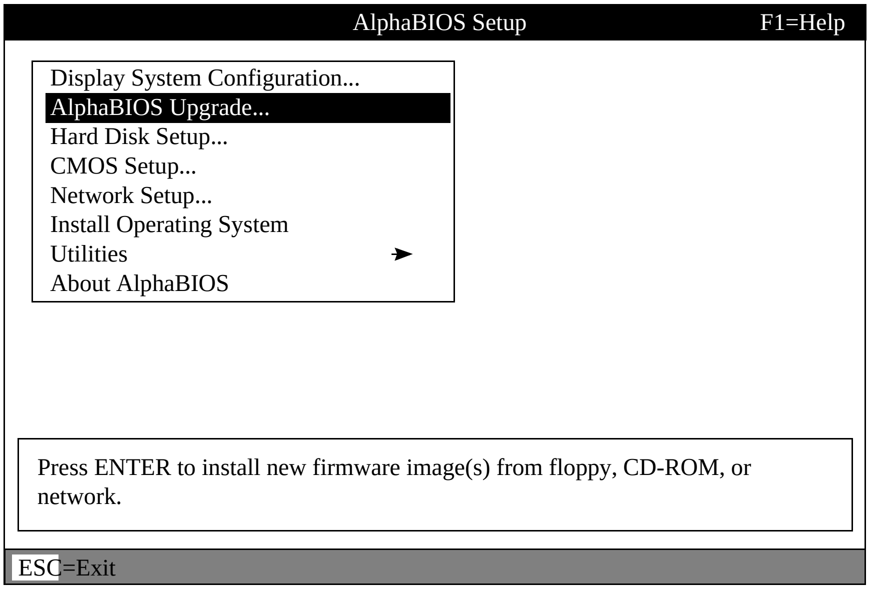
7.2.2 Upgrading AlphaBIOS

As new versions of Windows NT are released, it might be necessary to upgrade AlphaBIOS to the latest version. Also, as improvements are made to AlphaBIOS, you may wish to upgrade to take advantage of new features.

To upgrade from an earlier version of AlphaBIOS:

1. Insert the diskette or CD-ROM containing the AlphaBIOS upgrade.
2. If you are not already running AlphaBIOS Setup, restart your system and press **F2** when the boot screen is displayed.
3. In the main AlphaBIOS Setup screen, use the Tab key to select AlphaBIOS Upgrade and press **Enter** (see Figure 7–6). Then follow the instructions on the AlphaBIOS Upgrade screen.

Figure 7–6 Setup Screen Selecting AlphaBIOS Upgrade



Press ENTER to install new firmware image(s) from floppy, CD-ROM, or network.

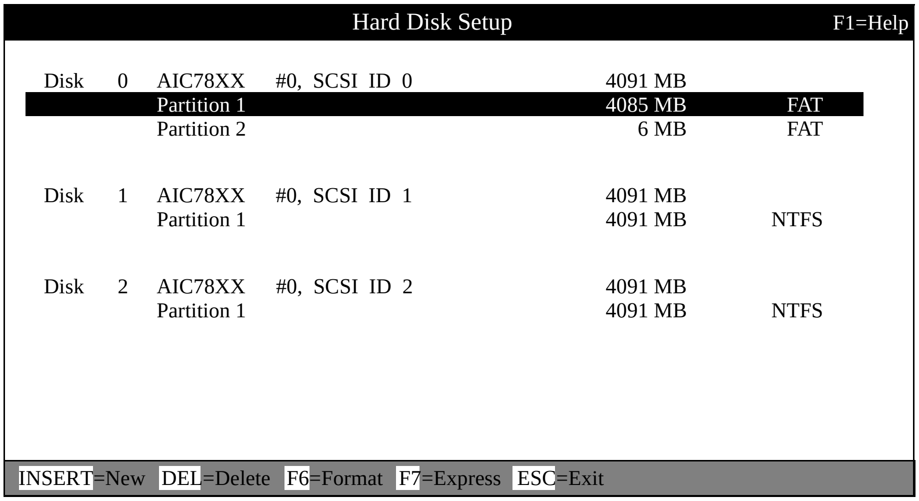
7.2.3 Setting Up Your Hard Disk

Express hard-disk setup is an easy way to create the recommended partition arrangement on your first hard disk, disk 0. Note, however, that express hard-disk setup does not format the large partition with NTFS. You will have the opportunity to do this during the Windows NT installation. Figure 7–7 shows an example of the Hard Disk Setup screen.

Note: AlphaBIOS assigns “disk 0” to the hard disk with the lowest SCSI ID number on the first bus in your system.

Caution: If you have any information on your disk that you want to save, be sure to back it up before using express hard-disk setup.

Figure 7–7 Hard Disk Setup Screen



```
                         Hard Disk Setup                              F1=Help

  Disk   0   AIC78XX    #0,  SCSI  ID  0              4091 MB
             Partition 1                              4085 MB          FAT
             Partition 2                                 6 MB          FAT


  Disk   1   AIC78XX    #0,  SCSI  ID  1              4091 MB
             Partition 1                              4091 MB          NTFS


  Disk   2   AIC78XX    #0,  SCSI  ID  2              4091 MB
             Partition 1                              4091 MB          NTFS






 INSERT=New  DEL=Delete  F6=Format  F7=Express  ESC=Exit
```

To perform express hard-disk setup:

1. If you have any needed information on your disk, back it up before proceeding.
2. Start AlphaBIOS Setup, select Hard Disk Setup, and press **Enter**.
3. Press **F7** to enter express hard-disk setup.
4. Press **F10** to continue with the setup.

For information on manually managing hard disks, see the online documentation.

7.2.4 Performing CMOS Setup Tasks

CMOS Setup is used to configure several system parameters and is divided into two modes, standard and advanced. Standard CMOS Setup is used to configure basic system parameters; Advanced CMOS Setup is used for system-specific parameters and password protection. Table 7–2 lists the specific tasks that you can perform in standard and advanced setup mode.

Table 7–2 CMOS Setup Tasks

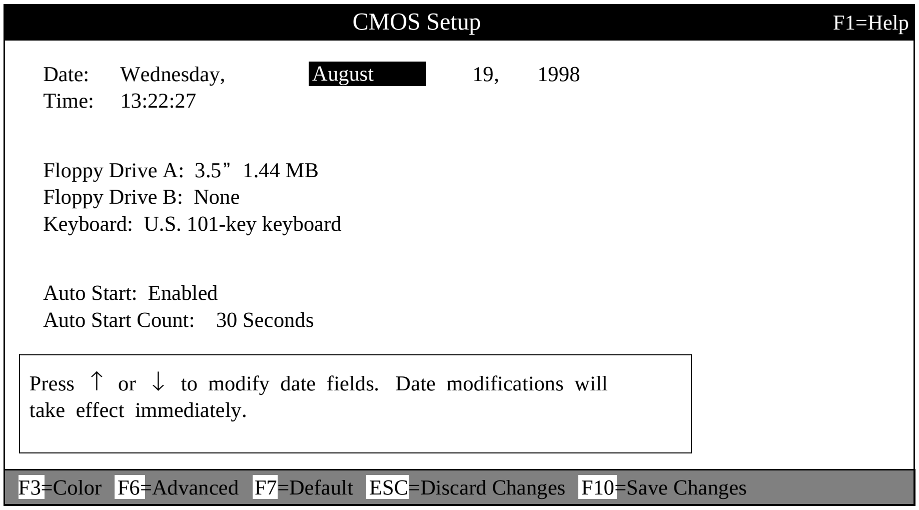
Standard CMOS Setup	Advanced CMOS Setup
Autostart	Memory test
Date and time	PCI parity
Floppy	Password protection
Keyboard	SCSI BIOS emulation
	S/G DMA mode
	Warm reboot

To enter standard CMOS Setup:

- Start AlphaBIOS Setup, select CMOS Setup, and press **Enter**.

Figure 7–8 shows a sample standard CMOS Setup screen.

Figure 7–8 Standard CMOS Setup Screen



CMOS Setup F1=Help

```
Date:    Wednesday,          August       19,    1998
Time:    13:22:27

Floppy Drive A:  3.5" 1.44 MB
Floppy Drive B:  None
Keyboard:  U.S. 101-key keyboard

Auto Start:  Enabled
Auto Start Count:   30 Seconds
```

Press ↑ or ↓ to modify date fields. Date modifications will take effect immediately.

F3=Color F6=Advanced F7=Default ESC=Discard Changes F10=Save Changes

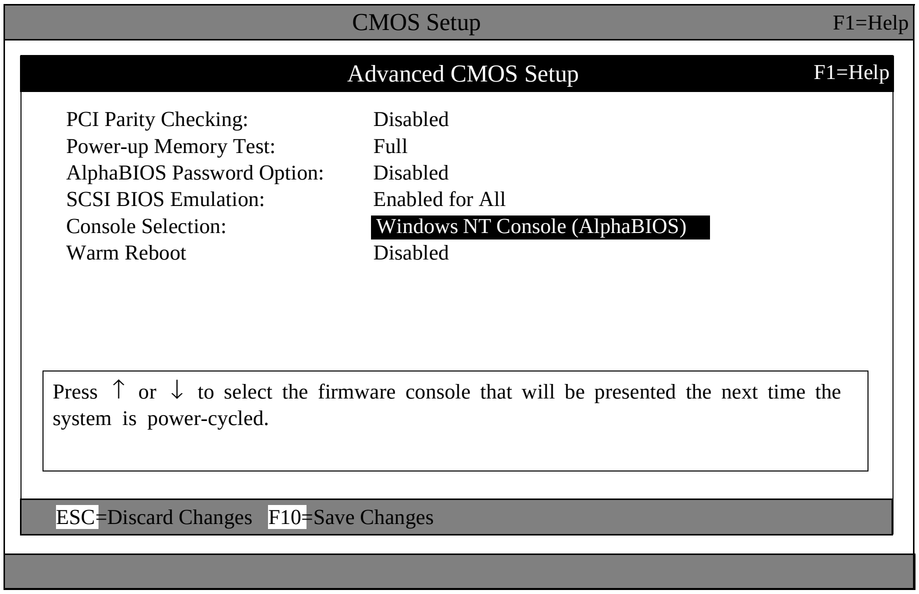
AlphaBIOS Setup Program

To enter advanced CMOS Setup:

1. Start AlphaBIOS Setup, select CMOS Setup, and press **Enter**.
2. In the CMOS Setup screen, press **F6**.

Figure 7–9 shows a sample advanced CMOS Setup screen.

Figure 7–9 Advanced CMOS Setup Screen



```
PCI Parity Checking:            Disabled
Power-up Memory Test:           Full
AlphaBIOS Password Option:      Disabled
SCSI BIOS Emulation:            Enabled for All
Console Selection:              Windows NT Console (AlphaBIOS)
Warm Reboot                     Disabled
```

Press ↑ or ↓ to select the firmware console that will be presented the next time the system is power-cycled.

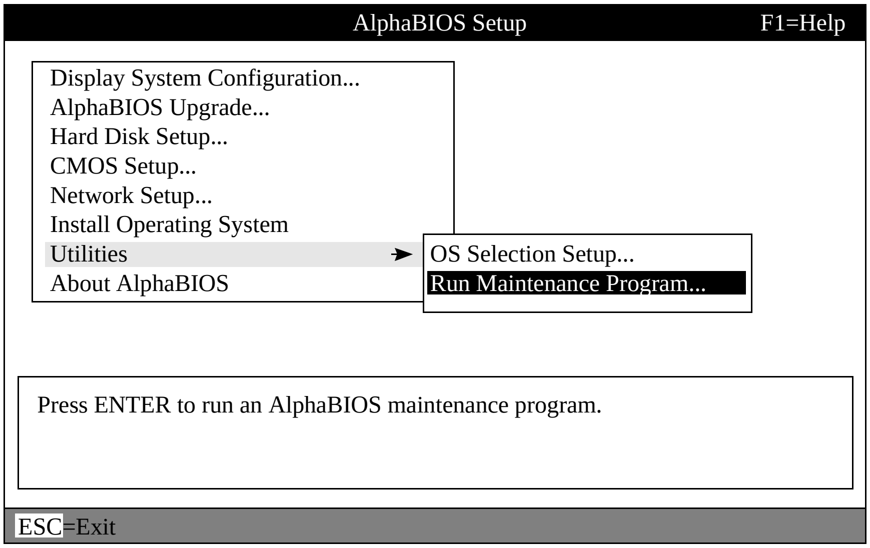
7.2.5 Running a Maintenance Program

You can run maintenance programs such as the RAID configuration utility (RCU) and other utilities directly from the AlphaBIOS utility menu.

To run a maintenance program:

1. Start AlphaBIOS Setup, select Utilities, select Run Maintenance Program from the submenu shown in Figure 7–10, and press **Enter**.

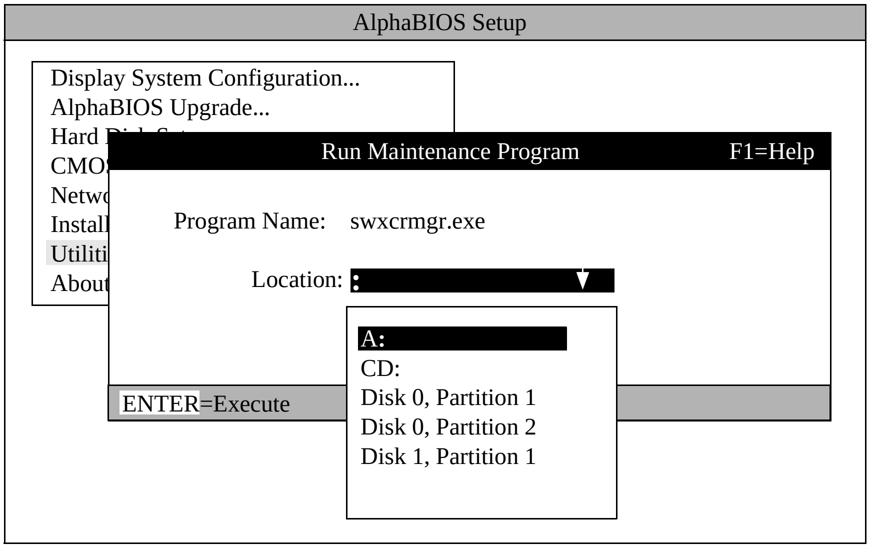
Figure 7–10 Run Maintenance Program Submenu



Press ENTER to run an AlphaBIOS maintenance program.

2. In the Run Maintenance Program dialog box, type the name of the program to be run at the Program Name prompt, as shown in Figure 7–11. Then tab to the Location listbox and select the hard-disk partition, floppy diskette, or CD-ROM drive from which to run the program.
3. Press **Enter** to execute the program.

Figure 7–11 Entering the Program Name



Program Name: swxcrmgr.exe

Location: :

If you are running a utility from a floppy diskette, you can simply type the utility's name into the Program Name field, and press **Enter**. This is because the floppy drive is the default selection in the Current Partition field.

Use Alt + ↓ when a listbox is selected to open the list.

7.3 Installing the Windows NT Operating System

This section supplements the Microsoft Windows NT installation guide for installing the Windows NT operating system on an AlphaPC 264DP system.

If you already have a previous version of Windows NT installed, see the Microsoft Windows NT installation guide for information about how to upgrade to a new version of the operating system.

7.3.1 Requirements

You need the following hardware and software to install the Windows NT operating system on an AlphaPC 264DP system:

- A 100MB or larger SCSI drive

Note: For information regarding supported SCSI hard drives, CD-ROM drives, and controllers supported as boot/installation devices, see the README.TXT file on the distribution medium.

- The Microsoft Windows NT compact disk
- The Installing Windows NT diskette you created

Note: A diskette is **required** for installing the Windows NT 4.0 operating system. Future versions of Windows NT may not require this diskette. Contact your OEM or system vendor for details.

If your system came with a Windows NT 4.0 installation diskette, you do not need to create a new diskette.

7.3.2 Before Installing Windows NT

Before installing Windows NT, ensure that you have completed the following tasks:

- Use the CMOS Setup program to:
  - Set the system date and time
  - Verify that all other standard CMOS settings match your system.

  Refer to Section 7.2.4 for more information about the CMOS Setup program.
- Perform an express hard-disk setup. This configures the first hard disk that AlphaBIOS detects in your system with the recommended default partition arrangement. See Section 7.2.3 for more information on setting up your hard disk. If you do not want to use the express hard-disk setup, refer to the online AlphaBIOS documentation for other hard-disk setup options.

7.3.3 Starting Windows NT Setup

Windows NT Setup, located on the Microsoft Windows NT compact disk, installs the Windows NT operating system. Windows NT Setup provides a brief description of the available key functions at the bottom of the screen.

During the Windows NT installation procedure, you may be prompted for the type of computer that you have. For the AlphaPC 264DP system, choose AlphaPC 264DP. If AlphaPC 264DP is not one of the choices, choose “other” *after* you have inserted your Windows NT installation diskette. The following procedure supersedes the Starting Setup on a RISC-based Computer section of the Microsoft Windows NT installation guide.

To start the Windows NT Setup program:

1. Insert the Windows NT compact disc in the CD-ROM drive.
2. Restart your computer.
3. Start AlphaBIOS Setup, use the Tab key to select Install Operating System, and press **Enter**.
4. Follow the instructions on the screen.

8

Alpha SRM Console Firmware

The Alpha SRM console firmware initializes the system and loads the AlphaBIOS firmware. This firmware resides in the flash ROM on the AlphaPC 264DP mainboard.

At power-up, the Alpha SRM console runs. If it is not configured to automatically load the AlphaBIOS, you may boot Tru64 UNIX.

8.1 Alpha SRM Console Firmware Conventions

The following conventions are used in this section:

Convention	Description
>>>	Alpha SRM console prompt
Backslash (\) at the end of a line	Continuation symbol to continue long commands on the next line
_>	Continuation line prompt
Maximum command length	255 characters
Multiple contiguous spaces or tabs	Treated as a single space
Command abbreviations	Allowed, if not ambiguous
Command qualifiers or options	Prefix with a space and a dash (-)
Numbers	Hexadecimal, unless otherwise specified (Registers, such as R0–R31, are shown in decimal notation.)

Alpha SRM Console Firmware Conventions

The following table lists Alpha SRM console special keys and their functions. These special keys, also referred to as shortcut keys, provide command recall, line editing, and basic input/output control flow.

Shortcut Key	Function
`Enter`	Terminate the command line input.
`Backspace or Delete`	Delete one character to the left of the cursor.
`Ctrl/A`	Toggles insert/overstrike mode. (Overstrike is the default.)

`Ctrl/B`
`Up arrow`
`Down arrow`
	Recall previous commands. (The last 16 commands are stored.)

`Ctrl/C`	Terminate the foreground process.

`Ctrl/D`
`Left arrow`
	Move the cursor one position to the left.

`Ctrl/E`	Move the cursor to the end of the line.

`Ctrl/F`
`Right arrow`
	Move the cursor one position to the right.

`Ctrl/H`	Move the cursor to the beginning of the line.
`Ctrl/O`	Suppress or resume (toggle) console output.
`Ctrl/Q`	Resume the flow (XON) of data to the console.
`Ctrl/R`	Retype the current command line.
`Ctrl/S`	Stop the flow (XOFF) of data to the console.
`Ctrl/U`	Delete the entire line.

8.2 Basic Alpha SRM Console Command Descriptions

This section describes the following basic Alpha SRM console commands that are necessary to boot the Tru64 UNIX operating system:

Command	Function
alphabios	Loads and starts the AlphaBIOS console
boot	Loads and starts Tru64 UNIX
deposit	Writes data to the specified address
edit	Invokes the console line editor on a RAM file or the nvram file
examine	Displays the contents of a memory location, register, or device
lfu	Loads and starts the Loadable Firmware Update utility
set	Configures operating system environment variables
show	Displays operating system environment variables or system information

The Alpha SRM console offers additional commands. For a complete list of Alpha SRM console commands, enter **help** at the Alpha SRM console prompt (>>>).

alphabios

Loads and runs the AlphaBIOS firmware, without changing the default firmware setting.

Syntax

```
alphabios
```

Arguments

None

Options

None

Description

The **alphabios** command starts the AlphaBIOS firmware. For more information on AlphaBIOS, see Chapter 7.

Example

The following **alphabios** command loads and runs AlphaBIOS:

```
>>>alphabios
```

Basic Alpha SRM Console Command Descriptions

boot

Initializes the processor, loads a program image from the specified boot device, and transfers control to the loaded image.

Syntax

```
boot [-file <filename>] [-flags
<longword>[,<longword>]]
[-protocols <enet_protocol>] [-halt]
[<boot_device>]
```

Arguments

`<boot_device>`
A device path or list of devices from which the firmware will attempt to boot. Use the **set bootdef_dev** command to set an environment variable that specifies a default boot device.

Options

boot Command Option	Description
-file <filename>	Specifies the name of a file to load into the system. Use the **set boot_file** command to set the environment variable that specifies a default boot file.
-flags <longword> [,<longword>]	Specifies additional information for the operating system. For Tru64 UNIX systems, the following values may be used: i = Interactive boot s = Boot to single user a = Autoboot to multiuser Use the **set boot_osflags** command to set an environment variable that specifies a default boot flag value.
-protocols <enet_protocol>	Specifies the Ethernet protocols that will be used for a network boot. Values may be **mop** or **bootp**.
-halt	Forces the bootstrap operation to halt and invoke the console program after the image is loaded and the page tables and other data structures are set up.

Description

The **boot** command initializes the processor, loads a program image from the specified boot device, and transfers control to that image. If you do not specify a boot device in the command line, the default boot device is used. The default boot device is determined by the value of the bootdef_dev environment variable.

If you specify a list of devices, a bootstrap is attempted from each device in the order in which the device is listed. Then control passes to the first successfully booted image. In a list, always enter network devices *last* because network bootstraps terminate only if a fatal error occurs or if an image is successfully loaded.

The **-flags** option can pass additional information to the operating system about the boot that you are requesting. If you do not specify a boot flag qualifier, the default boot flag's value specified by the boot_osflags environment variable is used.

The **-protocols** option allows selection of either the DECnet MOP or the TCP/IP BOOTP network protocols. The keywords **mop** and **bootp** are valid arguments for this option. It is possible to set the default protocol for a port by setting the environment variable ewa0_protocols or era0_protocols to the appropriate protocol.

Explicitly stating the boot flags or the boot device overrides the current default value for the current boot request, but does not change the corresponding environment variable.

See Section 8.4.1 for more information about environment variables.

Basic Alpha SRM Console Command Descriptions

Examples

boot Command Example	Description
`>>>boot`	Boots the system from the default boot device.
`>>>boot ewa0`	Boots the system from Ethernet port ewa0.
`>>>boot -file dec2.sys ewa0`	Boots the file named dec2.sys from Ethernet port ewa0.
`>>>boot -protocol bootp ewa0`	Boots the system using the TCP/IP BOOTP protocol from Ethernet port ewa0.
`>>>boot -flags 0,1`	Boots the system from the default boot device using flag setting 0,1.
`>>>boot -halt dka0`	Loads the bootstrap image from disk dka0, halts the bootstrap operation, and invokes the console program. Subsequently, you can enter **`continue`** to transfer control to the operating system.

deposit

Writes data to the specified address.

Syntax

```
deposit [-{b,w,l,q,o,h}] [{physical, virtual, gpr,
fpr, ipr}] [-n <count>] [-s <step>]
[<device>:]<address> <data>
```

Arguments

<device>:

The optional device name (or address space) selects the device to access. The following platform-independent devices are supported:

- pmem
  Physical memory.
- vmem
  Virtual memory. All access and protection checking occur. If the access is not allowed to a program running with the current processor status (PS), the console issues an error message. If memory mapping is not enabled, virtual addresses are equal to physical addresses.

`<address>`
An address that specifies the offset within a device into which data is deposited. The address may be any legal symbolic address.

Valid symbolic addresses are shown in the following table.

Symbolic Address	Description
gpr-name	Represents general-purpose register
ipr-name	Represents internal processor register
PC	Program counter
+	The location immediately following the last location referenced by **examine** or **deposit**

Symbolic Address	Description
-	The location immediately preceding the last location referenced by **examine** or **deposit**
*	The location last referenced by **examine** or **deposit**
@	The location addressed by the last location referenced by **examine** or **deposit**

`<data>`
The data to be deposited.

Options

deposit Command Option	Description
-b	Specifies data type is byte.
-w	Specifies data type is word.
-l	Specifies data type is longword.
-q	Specifies data type is quadword.
-o	Specifies data type is octaword.
-h	Specifies data type is hexword.
-physical	References physical address space.
-virtual	References virtual address space.
-gpr	References general-purpose register address space.
-fpr	References floating-point register address space.
-ipr	References internal processor register address space.
-n <count>	Specifies the number of consecutive locations to examine.
-s <step>	Specifies the address increment as a hexadecimal value. This option allows you to override the increment that is normally derived from the data size.

Description

The **deposit** command writes data to the specified address, such as a memory location, register, device, or file. The defaults for address space, data size, and address are the last specified values. After initialization, the default for address space is physical memory; for data size, the default is a quadword; and for address, the default is zero.

An address or device can be specified by concatenating the device name with the address. For example, use **pmem:0** and specify the size of the address space to be written. If a conflicting device, address, or data size is specified, the console ignores the command and issues an error response.

Examples

deposit Command Example	Description
`>>>d -n 1ff pmem:0 0`	Clears the first 512 bytes of physical memory
`>>>d -l -n 3 pmem:1234 5`	Writes the value 5 into four longwords, starting at physical memory address 1234
`>>>d -n 8 r0 ffffffff`	Loads GPRs R0 through R8 with -1
`>>>d -l -n 10 -s 200 pmem:0 8`	Writes the value 8 in the first longword of the first 17 pages in physical memory

edit

Loads and runs the Alpha SRM line editor.

Syntax

```
edit <file>
```

Arguments

`<file>`

Specifies the file to be edited.

Options

None

Description

The **edit** command starts the Alpha SRM line editor. For more information on the line editor, see Section 8.8.

Example

The following **edit** command starts the Alpha SRM line editor to edit the file nvram:

```
>>>edit nvram
```

examine

Displays the contents of the specified address.

Syntax

```
examine [-{b,w,l,q,o,h,d}] [-{physical, virtual,
gpr, fpr, ipr}] [-n <count>] [-s <step>]
[<device>:]<address>
```

Arguments

`<device>:`
The optional device name (or address space) selects the device to access.

`<address>`
The address specifies the first location to examine within the current device. The address can be any legal address specified.

Options

examine Command Option	Description
-b	Specifies data type is byte.
-w	Specifies data type is word.
-l	Specifies data type is longword.
-q	Specifies data type is quadword.
-o	Specifies data type is octaword.
-h	Specifies data type is hexword.
-d	Specifies the data displayed is the decoded macro instruction. The Alpha instruction decode (-d) does not recognize machine-specific PALcode instructions.
-physical	References physical address space.
-virtual	References virtual address space.
-gpr	References general-purpose register address space.
-fpr	References floating-point register address space.

examine Command Option	Description
-ipr	References internal processor register address space.
-n <count>	Specifies the number of consecutive locations to examine.
-s <step>	Specifies the address increment as a hexadecimal value. This option allows you to override the increment that is normally derived from the data size.

Description

The **examine** command displays the contents of the specified address, such as a memory location, register, device, or file. The defaults for address space, data size, and address are the last specified values. After initialization, the default for address space is physical memory; for data size, the default is a quadword; and for address, the default is zero.

An address or device can be specified by concatenating the device name with the address. For example, use **pmem:0** and specify the size of the address space to be displayed. If a conflicting device, address, or data size is specified, the console ignores the command and issues an error response.

The display line consists of the device name, the hexadecimal address (or offset within the device), and the examined data (also in hexadecimal).

The **examine** command supports the same options as the **deposit** command. Additionally, the **examine** command supports instruction decoding with the **-d** option, which disassembles instructions beginning at the current address.

Examples

examine Command		
Example	Display	Description
>>>e r0	`gpr:  0 (R0)  0000000000000002`	Examines the contents of R0, using a symbolic address.
>>>e -g 0	`gpr:  0 (R0)  0000000000000002`	Examines the contents of R0, using address space.
>>>e grp:0	`gpr:  0 (R0)  0000000000000002`	Examines the contents of R0, using a device name.

>>>examine -n 5 r7	`gpr:  38  (R7)  0000000000000000`
	`gpr:  40  (R8)  0000000000000000`
	`gpr:  48  (R9)  0000000000000000`
	`gpr:  50  (R10) 000000007FFBF800`
	`gpr:  58  (R11) 000000007FF781A2`
	`gpr:  60  (R12) 0000000000000000`
		Examines the contents of R7 and the next five registers.

>>>examine ipr:11	`ipr  11  (KSP) FFFFFFFF8228DFD0`	Examines the contents of internal processor register 11.

lfu

Loads and runs the Loadable Firmware Update (LFU) utility.

Syntax

```
lfu
```

Arguments

None

Options

None

Description

The **lfu** command starts the Loadable Firmware Update (LFU) utility. For more information on the LFU, see Section 6.5.

Example

The following **lfu** command loads and runs the LFU utility:

```
>>>lfu
```

set

Sets or modifies the value of an environment variable.

Syntax

```
set <envar> <value> [-default] [-integer] [-string]
```

Arguments

`<envar>`

The environment variable to be assigned a new value.

`<value>`

The value that is assigned to the environment variable. It can be either a numeric value or an ASCII string.

Options

set Command Option	Description
-integer	Creates an environment variable as an integer.
-string	Creates an environment variable as a string.

Description

The **set** command is used to set or modify the value of an environment variable. Environment variables are used to pass configuration information between the console and the operating system. See Section 8.4 for more information about environment variables.

Examples

set Command Example	Description
`>>>set bootdef_dev ewa0`	Modifies the default boot device to ewa0
`>>>set auto_action boot`	Attempts to boot the operating system following an error, halt, or power-up
`>>>set boot_osflags 0,1`	Modifies the default boot flags to 0,1
`>>>set foobar 5`	Creates an environment variable called foobar and gives it a value of 5

show

Displays the current value of the specified environment variable or information about the system.

Syntax

```
show [{config, device [device_name], iobq, hwrpb,
map, memory, pal, version, <envar>...}]
```

Arguments

show Command Argument	Description
config	Displays the current memory configuration, PCI logical slots, and ISA logical slots.
device [device name]	Displays the devices and controllers in the system. Specifying a device name returns information on that device only.
iobq	Displays the input/output counter blocks.
hwrpb	Displays the hardware restart parameter block.
map	Displays the system virtual memory map.
memory	Displays the memory module configuration.
pal	Displays the version of PALcode.
version	Displays the version of the console.
<envar>	Displays the current value of a specified environment variable.

Options

None

Description

The **show** command displays information about the system and the current value of a specified environment variable. See Section 8.4 for more information about environment variables.

Examples

show Command Example	Description
>>>**show device** dka0.0.0.6.0 DKA0 RZ26L 441A dka400.4.0.6.0 DKA400 RRD43 3213 dva0.0.0.0.1 DVA0 ewa0.0.0.12.0 EWA0 08-00-2B-E2-1C-25 pka0.7.0.6.0 PKA0 SCSI Bus ID 7	Lists device information, such as system designation, drive model, or Ethernet address
>>>**show memory** 48 Meg of System Memory	Lists system random-access memory (RAM) size
>>>**show *** (refer to Section 8.4)	Lists all environment variables and their settings
>>>**show boot*** (refer to Section 8.4)	Lists all environment variables, beginning with boot

8.3 Displaying System Parameters

System parameters can be displayed by using the **show** command with the appropriate parameter. Examples of the system configuration, CPUs, system devices, memory configuration, PALcode revision, and console revision are shown in the following subsections.

Note: In the examples, user input is underlined. Everything else is system output.

8.3.1 Displaying the System Configuration

The **show config** command displays a list of devices found on the system bus and I/O buses. The syntax is: *show config*. An example of the **show config** command is shown in Figure 8–1.

Figure 8–1 Example of Show Config Command

```
P00>>> show config
SRM console:    X5.3-1
PALcode:      OpenVMS PALcode V1.31-27, Tru64 UNIX PALcode V1.27-31
Processors
CPU 0           Alpha 21264-2 500 MHz      SROM Revision:  X1.70
                Bcache size: 2 MB
CPU 1           Alpha 21264-2 500 MHz      SROM Revision:  X1.70
                Bcache size: 2 MB

Core Logic
Cchip           DECchip 21272-CA Rev 2
Dchip           DECchip 21272-DA Rev 2
Pchip 0         DECchip 21272-EA Rev 2
Pchip 1         DECchip 21272-EA Rev 2

MEMORY
Array #       Size     Base Addr
-------    ----------  ---------
   2         512 MB    000000000
   3         512 MB    020000000

Total Bad Pages = 0
Total Good Memory = 1024 Mbytes
PCI Hose 00
     Bus 00  Slot 05/0: Cypress 82C693     Bridge to Bus 1, ISA
     Bus 00  Slot 05/1: Cypress 82C693  IDE
                            dqa.0.0.105.0
                            dqa0.0.0.105.0  TOSHIBA CD-ROM XM-56
Bus 00  Slot 05/2: Cypress 82C693 IDE
                           dqb.0.1.205.0
     Bus 00  Slot 05/3: Cypress 82C693 USB
     Bus 00  Slot 06/0: Adaptec AIC-7895
                           pkb0.7.0.6.0     SCSI Bus ID 7
     Bus 00  Slot 06/1: Adaptec AIC-7895
                           pkc0.7.0.106.0   SCSI Bus ID 7
     Bus 00  Slot 08: DECchip 21040-AA
                           ewa0.0.0.8.0     08-00-2B-E2-9C-60
     Bus 00  Slot 09: DEC PowerStorm
PCI Hose 01
     Bus 00  Slot 07: QLogic ISP1020
                           pka0.7.0.7.1   SCSI Bus ID 7
                           dka0.0.0.7.1   RZ28D
                           dka100.1.0.7.1 SEAGATE ST32155W
                           dka200.2.0.7.1 RZ28D
                           dka400.4.0.7.1 PLEXTOR CD-ROM PX-12
ISA
Slot  Device   Name       Type       Enabled  BaseAdd IRQ  DMA
0
        0      MOUSE      Embedded   Yes      60       12
        1      KBD        Embedded   Yes      60       1
        2      COM1       Embedded   Yes      3f8      4
        3      COM2       Embedded   Yes      2f8      3
        4      LPT1       Embedded   Yes      3bc      7
        5      FLOPPY     Embedded   Yes      3f0      6    2
P00>>>
```

8.3.2 Displaying the Number of CPUs

The **show cpu** command displays the status of each CPU. The syntax is: *show cpu.*

Figure 8–2 Example of Show CPU Command

```
P00>>> show cpu

Primary CPU:      00
Active CPUs:      00    01
Configured CPUs:  00    01
SROM Revision:    X1.72  X1.72

P00>>>
```

8.3.3 Displaying System Devices

The **show device** command displays the status for devices and controllers in the system: SCSI and MSCP devices, the internal floppy drive, and the network. The syntax is: *show device [controller_name]*, where [controller_name] indicates the controller name or an abbreviation. When abbreviations or wildcards are used, all controllers that match the type are displayed. If no name is given, the display is a list of all devices and controllers in the system.

An example of a device name is **dka200.2.0.7.1**. Table 8–1 shows the interpretation of this device name.

Table 8–1 Device Naming Convention

Category		Description			
dk	Driver ID	Two-letter designator of port or class driver.			
		dk	SCSI disk	fw	FDDI device
		dq	ATAPI CD-ROM	mk	SCSI tape
		dr	RAID set device	mu	DSSI tape
		du	DSSI disk	pk	SCSI port
		dv	Diskette drive	pu	DSSI port
		ew	Ethernet port		
a	Storage adapter ID	One-letter designator of storage adapter (a, b, c, ...).			
200	Device unit number	Unique number (MSCP init number). SCSI unit numbers are forced 100 × node ID.			
2	Bus node number	Bus node ID.			

Table 8–1 Device Naming Convention (Continued)

Category		Description
0	Channel number	Used for multi-channel devices.
7	Logical slot number	Corresponds to PCI slot number.
1	Hose number	0 for PCI0, 1 for PCI1.

Figure 8–3 Example of Show Device Command

```
P00>>> show device

dka0.0.0.7.1             DKA0     RZ28D  0010
dka100.1.0.7.1           DKA100   SEAGATE ST32155W  0596
dka200.2.0.7.1           DKA200   RZ28D  0010
dka400.4.0.7.1           DKA400   PLEXTOR CD-ROM PX-12TS  1.02
dqa0.0.0.105.0           DQA0     TOSHIBA CD-ROM XM-5602B 1156
dva0.0.0.0.0             DVA0
ewa0.0.0.8.0             EWA0     08-00-2B-E2-9C-60
pka0.7.0.7.1             PKA0     SCSI Bus ID 7  5.54
pkb0.7.0.6.0             PKB0     SCSI Bus ID 7
pkc0.7.0.106.0           PKC0     SCSI Bus ID 7
P00>>>
```

8.3.4 Displaying the Memory Configuration

The **show memory** command displays information about each memory bank: array number, size in megabytes, and the starting address. The syntax is: *show memory.*

Figure 8–4 Example of Show Memory Command

```
P00>>> show memory

Array #       Size     Base Addr
-------    ----------  ---------
   0         128 MB    000000000
   1         128 MB    008000000
   2         128 MB    010000000
   3         128 MB    018000000

Total Bad Pages = 0
Total Good Memory = 512 MBytes
P00>>>
```

8.3.5 Displaying PALcode Revision Information

The **show pal** command displays the versions of Tru64 UNIX PALcode. PALcode is the Alpha Privileged Architecture Library code, written to support Alpha processors. It implements architecturally defined processor behavior. The syntax is: *show pal*. Figure 8–5 shows an example of the **show pal** command.

Figure 8–5 Example of Show PAL Command

```
P00>>> show pal

pal  Tru64 UNIX PALcode V1.27-31
P00>>>
```

8.3.6 Displaying Console Revision Information

The **show version** command displays the version of the SRM console program that is installed on the system. The syntax is: *show version.* Figure 8–6 shows an example of the **show version** command.

Figure 8–6 Example of Show Version Command

```
P00>>> show version

version                                V5.3-1 JUL 20 1998 13:59:28
P00>>>
```

8.4 Environment Variables

Environment variables pass configuration information between the console and the operating system. Their settings determine how the system powers up, boots the operating system, and operates.

Environment variables are set or changed with the **set envar** command. Their values are viewed with the **show envar** command. Issue an **init** command to activate a new environment variable. Figure 8–7 shows an example of the **set envar** and **show envar** commands.

User-defined nonvolatile environment variables can be created with the **edit** command, described in Section 8.8. Section 8.4.1 describes the environment variables in detail.

Figure 8–7 Example of Set Envar and Show Envar Commands

```
P00>>> show console
console                  graphics
P00>>> set console serial
P00>>> show console
console                  serial
P00>>> init
```

set *envar*

The **set** command sets or modifies the value of an environment variable. It can also be used to create a new environment variable if the name used is unique. Environment variables are used to pass configuration information between the console and the operating system. The setting of these variables determines how the system powers up, boots the operating system, and operates. The syntax is:

set *envar value*

envar The name of the environment variable to be modified

value The new value of the environment variable

Whenever you modify the value of any of the following environment variables, the new value takes effect only after you reset the system by pressing the reset button or issuing the **initialize** command:

console
kbd_hardware_type
language
os_type

show *envar*

The **show envar** command displays the current value (or setting) of an environment variable. The syntax is:

show *envar*

envar The name of the environment variable to be displayed. The wildcard * displays all environment variables.

8.4.1 Environment Variable Summary

Environment variables are classified as either Alpha SRM console architecture-required or system-defined. Table 8–2 shows common Alpha SRM console architecture-required environment variables and their descriptions, and Table 8–3 shows common system-defined variables and their descriptions.

For a complete list, enter **show *** at the Alpha SRM console prompt.

Table 8–2 Architecture-Required Environment Variables

Variable	Description
`auto_action`	When used with the **set** or **show** command, this variable modifies or displays the console action that follows an error, halt, or power-up. The action can be halt, boot, or restart. The default is halt.
`boot_osflags`	When used with the **set** or **show** command, this variable modifies or displays the additional parameters to be passed to system software. The default is 0. The parameter choices are: **a** Load operating system software from the specified boot device (autoboot). Boot to multi-user mode. **i** Prompt for the name of file to load and other options (boot interactively). Boot to single-user mode. **s** Stop in single-user mode. Boots to single-user mode and stops at the # prompt. **D** Full dump; implies “s” as well. By default, if Tru64 UNIX crashes, it completes a partial memory dump.
`bootdef_dev`	When used with the **set** or **show** command, this variable modifies or displays the default device or device list from which the system will attempt to boot. If the system software is preloaded, the variable is preset to point to the device containing the preloaded software. The default is null. To specify more than one device with the **set** command, separate the devices with commas.

Table 8–3 System-Defined Environment Variables

Variable	Description
com*x*_baud	When used with the **set** command, this variable modifies the system baud rate to match that of the device connected to the port. The default is 9600.
console	When used with the **set** command, this variable modifies the console output to either the serial port or the graphics controller.
cpu_enabled	When used with the **set** command, this variable sets a bit mask that enables or disables specific CPUs on a multiprocessor system. Bit 0 of the mask corresponds to CPU0 and bit 1 to CPU1. A 0 in the bit mask prevents the CPU from running; a 1 allows it to run. The bit mask is a hexadecimal value.
ew*x*0_mode	When used with the **set** command, this variable specifies whether the AUI, ThinWire, twisted-pair, or fast Ethernet ports will be enabled.
ew*x*0_protocols	When used with the **set** command, this variable enables network protocols for booting and other functions. The parameter choices are: **mop** Sets the network protocol to mop (Maintenance Operations Protocol) (default). **bootp** Sets the network protocol to bootp, the setting typically used with Tru64 UNIX. **bootp,mop** When both are listed, the system attempts to use mop, regardless of which is listed first. If unsuccessful, the system uses bootp.
kbd_hardware_type	When used with the **set** command on systems with the language variant 3C (Français), this variable sets the keyboard hardware type as either PCXAL or LK411 and enables the system to interpret the terminal keyboard layout correctly. PCXAL is the default.

Table 8–3 System-Defined Environment Variables (Continued)

Variable	Description
language n	When used with the **set** command, this variable assigns language n to the system (where n is the option number of a language listed in the menu that follows). Use the following procedure to select the language: 1. At the Alpha SRM console prompt, enter the following commands: P00>>> set language 0 P00>>> init The following menu and prompt are displayed: n Language n Language ====================================== 0 none (display menu) 40 Français (Suisse Romande) 30 Dansk 42 Italiano 32 Deutsch 44 Nederlands 34 Deutsch (Schweiz) 46 Norsk 36 English (American) 48 Portugues 38 English (British/Irish) 4A Suomi 3A Español 4C Svenska 3C Français 4E Belgisch-Nederlands 3E Français (Canadian) (1..16): 2. Enter the number that corresponds to the language that you want to use. The following example shows how to assign the English (American) language to the system: (1..16): 36 3. When you receive a message to reset the system, power cycle the system.
memory_test	When used with the **set** command, this variable specifies the extent to which memory will be tested.
os_type	When used with the **set** command, this variable specifies the default power-up operating system interface. The parameter options are: nt AlphaBIOS is the default firmware. unix Alpha SRM console is the default firmware.

Table 8–3 System-Defined Environment Variables (Continued)

Variable	Description
pci_parity	When used with the **set** command, this variable controls PCI parity checking. The possible values are on or off (default).
pk*x*0_fast	When used with the **set** command, this variable enables a fast SCSI device to perform in either standard or fast (default) mode. The parameter options are 0 (standard) or 1 (fast).
pk*x*0_host_id	When used with the **set** command, this variable allows you to change the SCSI bus controller device, which is, by default, bus node 7.
pk*x*0_soft_term	When used with the **set** command, this variable enables/disables SCSI terminators. This command is used only when systems have the QLogic ISP1020 SCSI controller. The ISP1020 implements the 16-bit-wide SCSI bus and has two terminators: one for the low 8 bits, and one for the high 8 bits. The parameter options are: off Disables termination of all 16 bits low Enables low 8 bits termination and disables high 8 bits (default) high Enables high 8 bits termination and disables low 8 bits on Enables termination of all 16 bits diff Places the bus in differential mode
tt_allow_login	When used with the **set** command and the environment variable console is set to serial (COM1 port), this variable enables/ disables login to the SRM console firmware from alternate console ports. The parameter options are: 0 Disables login through the COM2 port or a graphics monitor 1 Enables login through the COM2 port or a graphics monitor (default)

8.5 Initializing the System

The **initialize** command resets the system and executes the power-up tests. Issuing this command is equivalent to pressing the Reset button. The syntax is *initialize* or *init.*

After self-tests are executed, the system autoboots unless one of the following conditions is true:

- A halt assertion condition exists (that is, the halt button has been pressed).
- The auto_action environment variable is set to halt.

If the auto_action environment variable is set to boot or restart and no halt assertion condition exists, the system autoboots. In all other cases, the system stops in console mode and does not attempt to boot.

Figure 8–8 shows an example of the **initialize** command.

Figure 8–8 Example of Initialize Command

```
P00>>> init

Initializing...
1024 Meg of system memory
probing hose 1, PCI
bus 0, slot 7 -- pka -- QLogic ISP1020
probing hose 0, PCI
probing PCI-to-ISA bridge, bus 1
bus 0, slot 5, function 1 -- dqa -- Cypress 82C693 IDE
bus 0, slot 5, function 2 -- dqb -- Cypress 82C693 IDE
bus 0, slot 6, function 0 -- pkb -- Adaptec AIC-7895
bus 0, slot 6, function 1 -- pkc -- Adaptec AIC-7895
bus 0, slot 8 -- ewa -- DECchip 21040-AA
bus 0, slot 9 -- vga -- DEC PowerStorm

AlphaPC 264 450 MHz Console X5.2-105, May 29 1998 17:03:22
P00>>>
```

8.6 Making the System Secure

Placing the console in secure mode ensures that unauthorized persons cannot gain access to the system. The commands for console security are **set password**, **clear password**, and **set secure**. The **login** command turns off security features during the current console session.

8.6.1 Set Password Command

The **set password** command sets the console password for the first time or changes an existing password. It is necessary to set the password only if the system is going to operate in secure mode.

If a password has not been set and the **set password** command is issued, the console prompts for a password and verification. If a password has been set and the **set password** command is issued, the console prompts for the new password and verification, then prompts for the old password.

The password is unchanged if the validation password entered does not match the existing password in the NVRAM. The password length must be between 15 and 30 alphanumeric characters.

The syntax is *set password*.

Figure 8–9 shows some examples of the **set password** command.

Figure 8–9 Examples of Set Password Command

Example 1: Setting a Password for the First Time

```
P00>>> set password
Please enter the password: ****************
Please enter the password again: ****************
P00>>>
```

Note:
Passwords and validations are not echoed.

Example 2: Changing a Password

```
P00>>> set password
Please enter the password: ******************
Please enter the password again: ******************
Now enter the old password: ****************
P00>>>
```

Example 3: Setting a Password with Too Few Characters

```
P00>>> set password
Please enter the password: ***********
Password length must be between 15 and 30 characters
P00>>>
```

8.6.2 Set Secure Command

The **set secure** command may be used after setting a password. It enables secure mode without requiring a restart of the console. If the password has been set, the console will be secured and only a small subset of commands can be performed. If a password has not been set, you are prompted to do so. The syntax is *set secure*.

Secure Mode Functions

When the console is in secure mode, the only commands recognized are **boot**, **login**, **continue**, and **start**.

- The **boot** command does not accept command-line parameters in secure mode. The console boots using the environment variables stored in NVRAM (boot_file, bootdef_dev, boot_flags). After a successful boot, the console is secured if there is a valid password.
- The **start** and **continue** commands are valid on a secure console. After either command is executed, the console is secured if there is a valid password. This prevents an intruder from accessing the system.

Figure 8–10 shows some examples of the **set secure** command in various circumstances.

Figure 8–10 Examples of Set Secure Command

Example 1: Set Secure Before a Password Has Been Set

```
P00>>> set secure
Secure not set. Please set the password.
P00>>>
```

Example 2: Set Secure After a Password Has Been Set

```
P00>>> set secure
Console is secure. Please login.
P00>>> b dkb0
Console is secure - parameters are not allowed.
P00>>> login
Please enter the password: ******************
P00>>> b dkb0
(boot dkb0.0.0.3.1)
 .
 .
 .
```

Note:
Passwords are not echoed.

8.6.3 Login Command

The **login** command turns off the security features, enabling access to all SRM console commands during the current session. The system automatically returns to secure mode as soon as the **boot**, **continue**, or **start** command is entered or when the system is initialized.

When the **login** command is entered, the user is prompted for the current system password. If a password has not been set, a message is displayed indicating that there is no password in NVRAM. If a password has been set, this prompt is displayed:

```
Please enter the password:
```

If the password entered matches the password in NVRAM, when the prompt is redisplayed, the console is no longer in secure mode and all console commands can be performed. The syntax is *login*.

Note: If you enter the **login** command when a halt assertion exists, the command fails, even if you enter the correct password.

Figure 8–11 shows some examples of the **login** command in various circumstances.

Figure 8–11 Examples of Login Command

Example 1: Login Command With System Not in Secure Mode

```
P00>>> login
Secure not set. Please set the password.
P00>>>
```

Example 2: Login Command With System in Secure Mode

```
P00>>> login
Please enter the password: **********************
P00>>>
```

Note:
Passwords are not echoed.

Example 3: Login Command With Incorrect Password

```
P00>>> login
Please enter the password:  *******************
Invalid password
P00>>>
```

Forgotten Password

If you have forgotten the current password, perform the following steps to clear the password.

1. Enter the **login** command:

   ```
   P00>>> login
   Please enter the password:
   ```

2. At the enter password prompt, press the Halt button, then press Enter.

The password is now cleared and the console cannot be put into secure mode unless a new password is set.

8.6.4 Clear Password Command

The **clear password** command clears the password environment variable, setting it to zero. This command is used when you want access to all the SRM console commands, but the system is in secure mode. To use **clear password**, you must know the current password. The syntax is *clear password*.

To clear the password without knowing the current password, use the **login** command with the halt button, as described in Section 8.6.3.

Figure 8–12 shows examples of the **clear password** command.

Figure 8–12 Examples of Clear Password Command

Example 1: Clear Password Command

```
P00>>> clear password
Please enter the password: ***************
Password successfully cleared.
P00>>>
```

Note:
Passwords are not echoed.

Example 2: Clear Password Command With Incorrect Password

```
P00>>> clear password
Please enter the password: ********************
Console is secure
P00>>>
```

8.7 Examine and Deposit Commands

The **deposit** command stores data in a specified location. The **examine** command displays the contents of a memory location, a register, or a device.

deposit

The **deposit** command stores data in the location specified. If no options are given with a **deposit** command, the system uses the options from the preceding **deposit** command.

If the specified value is too large to fit in the data size listed, the console ignores the command and issues an error response. If the data is smaller than the data size, the higher order bits are filled with zeros.

The syntax is *deposit [qualifiers] address data*.

examine

The **examine** command displays the contents of a memory location, a register, or a device.

If no options are given with an **examine** command, the system uses the options from the preceding **examine** command. If conflicting address space or data sizes are specified, the console ignores the command and issues an error.

For data lengths longer than a longword, each longword of data should be separated by a space.

The syntax is *examine [qualifiers] address*.

Figure 8–13 shows examples of **deposit** commands, and Figure 8–14 shows examples of **examine** commands.

Examine and Deposit Commands

Figure 8–13 Examples of Deposit Command

```
P00>>> dep -b -n 1ff pmem:0 0          Note: Clear first 512 bytes of physical memory.

P00>>> d -l -n 3 vmem:1234 5           Note: Deposit 5 into four longwords starting
                                                   at virtual memory address 1234.

P00>>> d -n 8 r0 ffffffff              Note: Load GPRs R0 through R8 with -1.

P00>>> d -l -n 10 -s 200 pmem:0 8      Note: Deposit 8 in the first longword of the first
                                             17 pages in physical memory.

P00>>> d -l pmem:0 0                   Note: Deposit 0 to physical memory address 0.

P00>>> d + ff                          Note: Deposit FF to physical memory address 4.

P00>>> d scbb 820000                   Note: Deposit 820000 to SCBB.
```

Figure 8–14 Examples of Examine Command

```
P00>>> examine pc                      Note: Examine the program counter.
PC psr: 0 ( PC)  0000000000001170


P00>>> examine sp                      Note: Examine the stack pointer.
gpr: F0 ( R30)  0000000000072A60


P00>>> e -n 6 r4                       Note: Examine register R4 and the next six registers.
grp: 20 (  R4) 0000000000005000
grp: 28 (  R5) 000000000FFFE000
grp: 30 (  R6) 00000003F8000C00
grp: 38 (  R7) 0000000053F761AE
grp: 40 (  R8) 0000010000000000
grp: 48 (  R9) 00000003F7800100
grp: 50 ( R10) 00000000000C7FFC


P00>>> examine pmem:400EC              Note: Examine physical memory.
pmem:  400EC A49D0078A47D0070
```

8.8 Creating a Power-Up Script

The system comes with a special nonvolatile file named “nvram” that is stored in EEROM. Nvram is a user-created power-up script (set of commands) that is always invoked during the power-up sequence.

You can use the **edit** command to create the initial nvram script or modify an existing one. The script should include any commands you want the system to execute at power-up.

With **edit**, lines may be added, overwritten, or deleted. The syntax is *edit file,* where file is the name of the file to be edited.

The editing commands are:

help	Displays the brief help file
list	Lists the current file prefixed with line numbers
renumber	Renumbers the lines of the file in increments of 10
exit	Leaves the editor and closes the file, saving all changes
quit	Leaves the editor and closes the file without saving changes
nn	Deletes line number nn
nn text	Adds or overwrites line number nn with text

Note: It is possible to disable the system by editing the nvram script. For example, if you include the **initialize** command in the script, the system will go into an endless loop. To fix this, press the Halt button while the system is powering up. You can then edit the script to delete the offending command.

Figure 8–15 shows how to modify the user-created power-up script “nvram.” In this example, the script is edited to include a command that sets the Ethernet protocol for the network device, ewa0, to TCPIP/BOOTP.

Figure 8–15 Example of Editing the nvram Script

```
P00>>> edit nvram
editing ’nvram’ 0 bytes read in
*10 set ewa0 protocols bootp
*list
10  set ewa0_protocols bootp
*exit
27 bytes written out to nvram
P00>>> nvram
```

Note: This command executes the script.

To clear the script, enter line numbers without any text. This deletes the lines. Figure 8–16 shows an example of this.

Figure 8–16 Example of Clearing the nvram Script

```
P00>>> edit nvram
editing ’nvram’
27 bytes read in
*10
*exit
0 bytes written out to nvram
P00>>>
```

8.9 Starting and Stopping CPUs

The **halt** and **continue** commands are used to stop and continue a program on the specified CPU.

The **halt** (or **stop**) command stops program execution on a secondary CPU that is still running a booted program. The syntax is *halt processor_number* or *stop processor_number* where *processor_number* is the logical CPU number displayed by the **show cpu** command.

The **continue** command resumes program execution on the specified processor or on the primary processor if none is specified. The processor begins executing instructions at the address that is currently in the program counter (PC). The processor is not initialized.

The **continue** command is valid only if you have not disturbed the system state and if you halted the system by pressing the Halt button on the control panel. The syntax is *continue [&pn] [address]* where *&pn* specifies the processor (*n* is 0 or 1) and *address* provides the starting address of the program. Figure 8–17 gives an example of the **halt** and **continue** commands.

Figure 8–17 Example of Halt and Continue Commands

```
P00>>> halt 1

halted CPU 1

halt code = 1
operator initiated halt
PC = ffffffff8007cc68
P00>>> continue &p1

continuing CPU 1
```

Note: Some console commands, for example, **boot**, can alter the machine state so that program mode cannot be successfully resumed (unless you include **-***halt* in the **boot** command). If program mode cannot be resumed, reboot the operating system.

Other commands that alter machine state are **lfu**, **show device**, **show network**, **test**, and **start**.

8.10 Getting Help

The **help** (or **man**) command displays basic information about the use of console commands when the system is in console mode. The syntax is *help [command . . .]* where *command . . .* is the command or topic for which help is requested.

The options for *command . . .* are:

none — Displays the complete list of commands for which you can receive help

command_name — Displays information about the **console** command

argument_string (such as “sh”) — Displays information about all commands that begin with that string

Figure 8–18 shows an example of a **help** command.

Figure 8–18 Example of Help Command

```
P00>>> help set
NAME
              set
FUNCTION
              Set an option or modify the value of an environment
              variable.
SYNOPSIS
              set <option> <value> or <envar> [-] <value>
                    where
                    <option>={host,mode}
                    where
              <envar>={auto_action,bootdef_dev,boot_osflags,...}
                    [-default]
```

8.11 Upgrading the System

The **lfu** command is used to update firmware from the SRM console prompt. The **lfu** command starts the Loadable Firmware Update (LFU) utility. The syntax is *lfu*.

Note: If the system has been shut down from a booted program (most commonly, the operating system) or in some other way halted back to the SRM console, the system must be reset before running LFU.

Figure 8–19 shows an example of an **lfu** command. See Section 6.5 for more information about LFU.

Figure 8–19 Example of Lfu Command

```
P00>>>lfu
Checking dka400.4.0.7.1 for the option firmware files. . .
Checking dva0 for the option firmware files. . .

Option firmware files were not found on CD or floppy.
If you want to load the options firmware,
please enter the device on which the files are located(ewa0),
or just press <return> to proceed with a standard console update: dva0
Please enter the name of the options firmware files list, or
press <return> to use the default filename (pc264fw.txt)  : pc264fw.txt
Copying pc264fw.txt from dva0. . .
Copying PC264NT.ROM from dva0. . .
Copying PC264SRM.ROM from dva0. . .

                 ***** Loadable Firmware Update Utility *****
-----------------------------------------------------------------------------
Function    Description
-----------------------------------------------------------------------------
Display     Displays the system’s configuration table.
Exit        Done exit LFU (reset).
List        Lists the device, revision, firmware name, and update revision.
Readme      Lists important release information.
Update      Replaces current firmware with loadable data image.
Verify      Compares loadable and hardware images.
? or Help   Scrolls this function table.
-----------------------------------------------------------------------------

UPD> list
Device         Current Revision    Filename            Update Revision
nt                 5.68            nt_fw                   5.68
srm                5.4-4           srm_fw                  5.4-5
UPD> update *
Confirm update on:
nt
srm
[Y/(N)] y

WARNING: updates may take several minutes to complete for each device.

                          DO NOT ABORT!

nt              Updating to 5.68...  Verifying 5.68...  PASSED.
srm             Updating to 5.4-5...  Verifying 5.4-5...  PASSED.
UPD>
```

8.12 Booting the Operating System

The **boot** command is used to boot the operating system. It initializes the processor, loads a program image from the specified boot device, and transfers control to that image. The syntax is:
*boot [-**file** filename] [-**flags** [value]] [-**halt**] [-**protocols** enet_protocol] [boot_dev]*

The **boot** qualifiers and parameters can be the following:

-file filename — The boot file.

-flags *[value]* — Specifies additional information to the loaded image or operating system. In Tru64 UNIX, specifies boot flags. This qualifier overrides the setting of the *boot_osflags* environment variable. See the *boot_osflags* environment variable (Section 8.4.1) for a list of settings and their meanings.

-halt — Forces the bootstrap operation to halt and invoke the console program once the bootstrap image is loaded and page tables and other data structures are set up. Console device drivers are not shut down. Transfer control to the image by entering the **continue** command.

-protocols *enet_protocol* — Either mop (default) or bootp. This qualifier overrides the setting of the ew*0_protocols environment variable.

boot_dev — A device path or list of devices from which the console program attempts to boot, or a saved boot specification in the form of an environment variable. This qualifier overrides the setting of the *bootdef_dev* environment variable (see Section 8.4.1). Use the *bootdef_dev* environment variable to define the default boot device string.

Figure 8–20 shows an example of a **boot** command.

Figure 8–20 Example of Boot Command

```
P00>>> boot
(boot dka0.0.0.9.1 -flags A)
block 0 of dka0.0.0.9.1 is a valid boot block
reading 16 blocks from dka0.0.0.9.1
bootstrap code read in
base = 200000, image_start = 0, image_bytes = 2000
initializing HWRPB at 2000
initializing page table at 3ff78000
initializing machine state
setting affinity to the primary CPU
jumping to bootstrap code

Digital UNIX boot - Fri Nov 13 13:15:46 EST 1998

Loading vmunix ...
Loading at 0xfffffc0000230000
Current PAL Revision <0x2002b00010133>
Switching to OSF PALcode Succeeded
New PAL Revision <0x2002d0002012f>

Sizes:
text = 2981824
data = 452816
bss  = 1341040
Starting at 0xfffffc00003ea0e0

Alpha boot: available memory from 0x1800000 to 0x3ff76000
Digital UNIX V4.0E (Rev. 1091); Thu Jan 21 16:53:01 EST 1999
physical memory = 1024.00 megabytes.
available memory = 999.47 megabytes.
using 3922 buffers containing 30.64 megabytes of memory
Firmware revision: 5.4-6787
PALcode: Digital UNIX version 1.47-45
AlphaPC 264DP 500 MHz
pci1 at nexus
tu0: DECchip 21140: Revision: 2.2
tu0: auto negotiation capable device
tu0 at pci1 slot 8
tu0: DEC TULIP (10/100) Ethernet Interface, hardware address: 00-00-F8-
1B-0B-24
tu0: auto negotiation off: selecting 100BaseTX (UTP) port: half duplex
isp0 at pci1 slot 9
isp0: QLOGIC ISP1040B/V2
isp0: Firmware revision 5.57 (loaded by console)
isp0: Fast RAM timing enabled.
scsi0 at isp0 slot 0
rz0 at scsi0 target 0 lun 0 (LID=0) (DEC     RZ28D    (C) DEC 0010)
(Wide16)
rz1 at scsi0 target 1 lun 0 (LID=1) (SEAGATE ST34520W         1281)
(Wide16)
```

```
rz2 at scsi0 target 2 lun 0 (LID=2)  (DEC     RZ28D    (C) DEC 0010)
(Wide16)
gpc0 at isa0
pci0 at nexus
ace0 at isa0
ace1 at isa0
lp0 at isa0
fdi0 at isa0
fd0 at fdi0 unit 0
ata0 at pci0 slot 105
ata0: Cypress 82C693
scsi1 at ata0 slot 0
rz8 at scsi1 target 0 lun 0 (LID=3) (TOSHIBA CD-ROM XM-5602B  1156)
ata1 at pci0 slot 205
ata1: Cypress 82C693
scsi2 at ata1 slot 0
tga0 at pci0 slot 7
tga0: depth 8, map size 2MB, 1024x768
tga0: ZLXp2-E, Revision: 34
itpsa0 at pci0 slot 9
IntraServer ROM Version V1.0 (c)1998
scsi3 at itpsa0 slot 0
rz28 at scsi3 target 4 lun 0 (LID=4) (PLEXTOR CD-ROM PX-12TS  1.02)
Created FRU table binary error log packet
kernel console: ace0
dli: configured
Checking local filesystems
/sbin/ufs_fsck -p
/dev/rrz0a: 1557 files, 65618 used, 63229 free (421 frags, 7851 blocks,
0.3% fragmentation)
/dev/rrz0g: File system unmounted cleanly - no fsck needed
/dev/rrz0h: File system unmounted cleanly - no fsck needed
Mounting / (root)
user_cfg_pt: reconfigured
Mounting local filesystems
/dev/rz0a on / type ufs (rw)
/proc on /proc type procfs (rw)
/dev/rz0g on /usr type ufs (rw)
/dev/rz0h on /var type ufs (rw)
Subsystem xx was successfully configured.
Feb 12 13:40:22 update: started
The system is coming up. Please wait...
Checking for crash dumps
Initializing paging space
swapon: added /dev/rz0b as swap device.
swapon: added /dev/rz1c as swap device.
Mounting Memory filesystems
```

```
Streams autopushes configured
Configuring network
hostname: ebfw04.hlo.dec.com
Loading LMF licenses
System error logger started
Binary error logger started
route daemon started
Setting kernel timezone variable
Setting the current time and date with ntpdate
ONC portmap service started
NFS IO service started
Mounting NFS filesystems
Preserving editor files
security configuration set to default (BASE).
Successful SIA initialization

Clearing temporary files
Unlocking ptys
SMTP Mail Service started.
Network Time Service started
Extensible SNMP master agent started
Base O/S sub-agent started
Server System sub-agent started
Server Management sub-agent started
Environmental Monitoring Subsystem Configured.
Internet services provided.
Using snmp service entry port 161.
Cron service started
SuperLAT. Copyright 1994 Meridian Technology Corp. All rights reserved.
LAT started.
Printer service started
The system is ready.

Digital UNIX Version V4.0 (ebfw04.hlo.dec.com) console

login:
```

8.13 Installing the Tru64 UNIX Operating System

Note: Tru64 UNIX Version 4.0E is the minimum revision required for the AlphaPC 264DP mainboard system.

This section supplements the *DIGITAL UNIX Installation Guide* for installing the Tru64 UNIX operating system on an AlphaPC 264DP mainboard system.

Note: If you already have a previous version of Tru64 UNIX installed, see the *DIGITAL UNIX Installation Guide* for information on how to upgrade to a new version of the operating system. Be sure to also review the preinstallation tasks in this document.

8.13.1 Requirements

You need the following hardware and software to install the Tru64 UNIX operating system on an AlphaPC 264DP mainboard system:

- A minimum of 32MB of main memory; 64MB is recommended
- A 1GB (or larger) SCSI hard disk capable of storing the supported software subsets
- Supported load devices
  - SCSI CD–ROM drive capable of reliably reading in 512-byte block mode

    or
  - A network interface
- A console terminal with ASCII capability or a supported graphics display console
- Tru64 UNIX Version 4.0E or higher Operating System Volume 1 compact disc
- Alpha SRM console Version 5.4 or higher

9
Troubleshooting

This chapter contains information about troubleshooting hardware and software during AlphaPC 264DP startup.

9.1 Password Bypass

AlphaBIOS

AlphaBIOS provides password protection. However, password bypass is provided for system setup or startup when the AlphaBIOS password is unavailable.

Normal operation, with switch 8 (pby) of SW2 in the off position (see Figure 3–1), requires a password. When the password bypass function is enabled (moving switch 8 to the on position), it disables the AlphaBIOS password verification and enables the user to set up or start up their system without the AlphaBIOS password. Password bypass also clears the password.

After this function has been enabled, to disable it and require a password, set switch 8 to the off position.

Alpha SRM Console

Once the console has been put into secure mode, the Alpha SRM console requires a password to perform certain functions.

If the console is in secure mode and you have forgotten the current password, perform the following steps to clear the password:

1. Enter the login command.
2. At the **Enter Password** prompt, press the Halt button, then press Enter.

The password is now cleared and the console cannot be put into secure mode unless a new password is set.

9.2 Hardware Startup

Use the following troubleshooting steps if video is not working on your system. If you still have no video after reviewing these steps, please call your system vendor.

Troubleshooting Steps: No Video

1. Check the connection to the ac outlet.
2. Check the voltage setting on the power supply (115 V ac in the U.S.).
3. Check that the frequency/switch selection matches the speed of the Alpha chip.
4. Check that the CPU fan is connected and spinning.
5. Ensure that the flash ROM update procedure was performed correctly. If you have a terminal attached to COM1, check the output for error messages after verifying that the flash ROM update procedure was performed correctly.
6. Reseat the video card and ensure that it is connected to the monitor.
7. Reseat the DIMMs.
8. Replace the DIMMs.

9.3 Daughtercard LEDs

There are five LEDs located on the daughtercard to show the status of functions on the daughtercard. Table 9–1 shows the meaning of the LEDs.

Table 9–1 AlphaPC 264DP Daughtercard LEDs

LED	Function	Comments
D1	CPU self-test	Becomes asserted at completion of built-in self-test (BIST). If it does not assert, self-test did not complete.
D2	dc_ok	Asserted by the reset FPGA when it senses dc power.
D3	srom_clk	Fast flash (appears on but dim) while SROM is being loaded.
D4	2v_dcok	Asserted while 2 V to the CPU is present.
D5	5v_dcok	Asserted while 5 V to the CPU is present.

9.4 Beep Codes

The beep codes provide error information about the AlphaPC 264DP system. Table 9–2 lists and describes the beep codes.

Table 9–2 Beep Codes

Beep Code	Description
1–2–3[1]	This sequence represents the fail-safe booter startup.
4	No valid header found in ROM; loading entire ROM.
6	Memory error detected.

[1] One beep and a pause, followed by two beeps and a pause, followed by three beeps.

9.5 Post Codes

The post codes indicate the progress of the SROM and AlphaBIOS firmware. Table 9–3 lists and describes the post codes.

Note: The post codes are visible on an ISA-based postcard that displays codes written to ISA port 80.

Table 9–3 Post Codes

Source	Post Code (hex)	Description
SROM	20	Firmware initialization is complete.
	01	CPU speed detected; initialize southbridge.
	02	Southbridge ready.
	03	Initialize Bcache.
	05	Start sweep of Bcache and memory.
	06	Bcache and memory ready.
	14	Load system code.
	15	Loading uncompressed firmware into memory.
	16	Loading compressed firmware into memory.
	17	Jump to loaded firmware.
	3F	Fatal error. Second code identifies source of error: 06 = No memory found or bad memory.
AlphaBIOS	98	Firmware has control.
	97	Buses initialized.

Table 9–3 Post Codes (Continued)

Source	Post Code (hex)	Description
	95	Video console opened.
	51	FATAL - returned from boot menu selection.
	50	FATAL - firmware fatal error.
	42	Firmware drivers have been started.
	00	Firmware has been initialized.
Fail-safe booter	FD	I/O bus initialization.
	FC	I/O initialization complete.
	FB	UART initialization.
	BF	Fail-safe booter is scanning the diskette.
	1	Fail-safe booter is loading the Windows NT AlphaBIOS.
	2	Fail-safe booter is loading the Debug Monitor.
	3	Fail-safe booter is loading the SRM console.
	4	Fail-safe booter is loading the Linux mini-loader.

9.6 Fail-Safe Booter

The Fail-Safe Booter (FSB) utility provides an emergency recovery mechanism when the firmware image contained in flash memory has been corrupted. You can run the FSB and boot another image from a diskette that is capable of reprogramming the flash ROM. Refer to Section 6.3.

9.7 Windows NT Startup

While booting the Windows NT operating system, you may receive one of the following error messages:

- `The required system file DISPLAY_DRIVER.DLL is bad or missing.`
- `The required system file DISPLAY_DRIVER.SYS is bad or missing.`

These error messages indicate video driver problems. Use the following chart to solve common video driver problems.

If the...	Then...
Driver becomes disabled	Boot the Windows NT operating system with the last known good configuration.
Incorrect video driver is chosen while installing the Windows NT operating system	Reinstall the Windows NT operating system with the correct driver.
New video card was installed before the driver was installed	Reinstall the old video card, boot the Windows NT operating system, and install the driver for the new video card. Then install the new video card and reboot the Windows NT operating system.

A

Support, Products, and Documentation

A.1 Customer Support

Alpha OEM provides the following web page resources for customer support.

URL	Description
http://www.digital.com/alphaoem	Contains the following links: • **Developers’ Area:** Development tools, code examples, driver developers’ information, and technical white papers • **Motherboard Products:** Motherboard details and performance information • **Microprocessor Products:** Microprocessor details and performance information • **News:** Press releases • **Technical Information:** Motherboard firmware and drivers, hardware compatibility lists, and product documentation library • **Customer Support:** Feedback form

A.2 Supporting Products

This section lists sources for components and accessories that are not included with the AlphaPC 264DP.

A.2.1 Memory

Dual inline memory modules (DIMMs) are available from a variety of vendors. For a list of the qualified vendors, visit the Alpha OEM World Wide Web Internet site at URL:

http://www.digital.com/alphaoem

Click on **Technical Information**.
Then click on **Alpha OEM Hardware Compatibility List**.

A.2.2 Power Supply

A power supply, suitable for use with the AlphaPC 264DP (+3.3 V, +5 V, –5 V, +12 V, –12 V), is available from:

Antec, Inc.

2859 Bayview Drive
Fremont, CA 94538
Phone: 510–770–1200, ext. 312
PN PRS-618 (630 W)

A.2.3 Enclosure

An enclosure, suitable for housing the AlphaPC 264DP and its power supply, is available from:

Delta Axxion Technology

1550 Northwestern
El Paso, TX 79912
Phone: 915–877–5288
PN TL-22

A.3 Associated Documentation

You can order the following associated documentation directly from the vendor.

Title	Vendor
Alpha Architecture Reference Manual EY–W938E–DP	Call your sales office or call Butterworth-Heinemann (DIGITAL Press) at 1–800–366–2665.
Alpha Architecture Handbook EC–QD2KB–TE	Order online: **http://www.digital.com/alphaoem**
Alpha 21264 Microprocessor Hardware Specification DS–0013C–TE	See previous entry
PCI Local Bus Specification, Revision 2.1 PCI Multimedia Design Guide, Revision 1.0 PCI System Design Guide PCI-to-PCI Bridge Architecture Specification, Revision 1.0 PCI BIOS Specification, Revision 2.1	PCI Special Interest Group U.S. 1–800–433–5177 International 1–503–797–4207 Fax 1–503–234–6762
CY82C693UB hyperCache/Stand-Alone PCI Peripheral Controller with USB Data Sheet	Cypress Semiconductor Corp. 3901 North First Street San Jose, CA 95134 Phone: 1–800–858–1810
Super I/O Floppy Disk Controller with Infrared Support (FDC37C669) Data Sheet	Standard Microsystems Corporation 80 Arkay Drive Hauppauge, NY 11788 Phone: 1–800–443–7364 Fax: 1–516–231–6004