Digital Alpha VME 2100

Service Guide

Order Number: EK–DALPH–SG. A01

Digital Equipment Corporation
Maynard, Massachusetts

First Printing, April 1995

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

FCC NOTICE: The equipment described in this manual generates, uses, and may emit radio frequency energy. The equipment has been type tested and found to comply with the limits for a Class A computing device pursuant to Subpart J of Part 15 of FCC Rules, which are designed to provide reasonable protection against such radio frequency interference when operated in a commercial environment. Operation of this equipment in a residential area may cause interference, in which case the user at his own expense may be required to take measures to correct the interference.

S2781

This document was prepared using VAX DOCUMENT Version 2.1.

Contents

3 Running System Diagnostics

4 Error Log Analysis

5 System Configuration and Setup

6 Digital Alpha VME 2100 (BA742 Enclosure) FRU Removal and Replacement

Examples

Figures

Tables

Preface

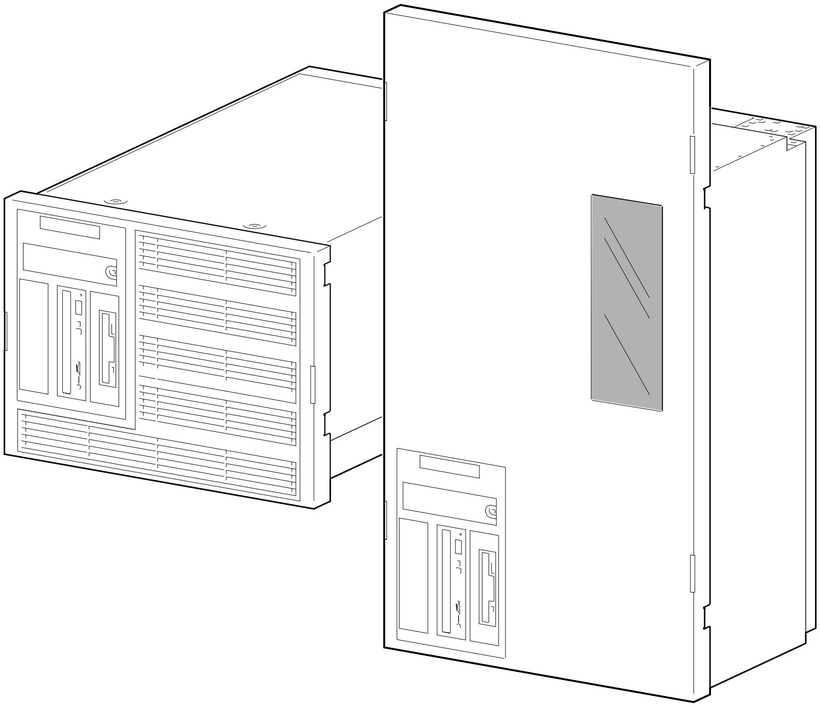
Purpose of this Guide

This guide describes the procedures and tests used to service Digital Alpha 2100 VME systems. These systems use two versions of the BA742 rackmount enclosure:

- Drawer-mount system
- Vertical-mount system

Figure 1 show the systems included in this guide.

Figure 1 Digital Alpha 2100 VME Systems



MLO-011591

Intended Audience

This guide is intended for use by Digital Equipment Corporation service personnel and qualified self-maintenance customers.

Conventions

The following conventions are used in this guide:

Convention	Meaning
Return	A key name enclosed in a box indicates that you press that key.
Ctrl/*x*	Ctrl/*x* indicates that you hold down the Ctrl key while you press another key, indicated here by *x*. In examples, this key combination is enclosed in a box, for example, Ctrl/C.
Warning	Warnings contain information to prevent personal injury.
Caution	Cautions provide information to prevent damage to equipment or software.
`boot`	Console and operating system commands are shown in special typeface.
`[ ]`	In command format descriptions, brackets indicate optional elements.
`show config`	Console command abbreviations must be entered exactly as shown. Commands shown in lowercase can be entered in either uppercase or lowercase.
italic type	Italic type in console command sections indicates a variable.
< >	In console mode online help, angle brackets enclose a placeholder for which you must specify a value.
{ }	In command descriptions, braces containing items separated by commas imply mutually exclusive items.

Related Documentation

The following information lists related system documents that supplement this guide.

- *Digital Alpha VME 2100 Owner's Guide*, EK–DALPHP–OG
- *Digital Alpha VME 2100 Vertical Rack Mount Front Bezel*, EK–DALVR–IS
- *Digital Alpha VME 2100 Drawer Rack Mount Front Bezel*, EK–DALDR–IS
- *AlphaServer 2000/2100 Firmware Reference Guide*, EK–AXPFW–RM
- *DEC Verifier and Exerciser Tool User's Guide*, AA–PTTMA–TE
- *Guide to Kernel Debugging*, AA–PS2TA–TE
- *OpenVMS Alpha System Dump Analyzer Utility Manual*, AA–PV6UB–TE
- *DECevent Translation and Reporting Utility for OpenVMS User and Reference Guide*

1 Troubleshooting Strategy

This chapter describes the troubleshooting strategy for Digital Alpha 2100 VME systems.

- Section 1.1 provides questions to consider before you begin troubleshooting an Digital Alpha 2100 VME system.
- Tables 1–1 through 1–5 provide a diagnostic flow for each of the categories of system problems.
- Section 1.2 lists the product tools and utilities.
- Section 1.3 lists available information services.

1.1 Troubleshooting the System

Before troubleshooting any system problem, check the site maintenance log for the system's service history. Be sure to ask the system manager the following questions:

- Has the system been used before and did it work correctly?
- Have changes to hardware or updates to firmware or software been made to the system recently?
- What is the state of the system—is the operating system running?

  If the operating system is down and you are not able to bring it up, use the console environment diagnostic tools, such as the power-up/diagnostic displays and ROM-based diagnostics (RBDs).

  If the operating system is running, use the operating system environment diagnostic tools, such as error logs, crash dumps, and exercisers (DEC VET).

1.1.1 Problem Categories

System problems can be classified into five categories listed below and in the following tables. Using these categories, you can quickly determine a starting point for diagnosis and eliminate the unlikely sources of the problem.

1. Power problems (Table 1–1)
2. No access to console mode (Table 1–2)
3. Console-reported failures (Table 1–3)
4. Boot failures (Table 1–4)
5. Operating system-reported failures (Table 1–5)

Table 1–1 Diagnostic Flow for Power Problems

Symptom	Action
Power supply fan does not spin up when the AC power cable is plugged into the power supply.	Check the power source and power cord.
AC power is present, as indicated by spinning fan, but system does not power on.	Check the DC On/Off button setting on the operator control panel.
	Check that the ambient room temperature is within environmental specifications (10–40°C, 50–104°F).
	Check that internal power supply cables are plugged in at both the power supply and motherboard.
Power supply shuts down after approximately 5 seconds (fan failure).	Check to see if both 6.75-inch fans are operating. A failure of either 6.75-inch fan causes the system to shut down after approximately 5 seconds.

Table 1–2 Diagnostic Flow for Problems Getting to Console Mode

Symptom	Action
Power-up screen is not displayed.	Check power-up/diagnostic display on the OCP (Section 2.1) for a failure during self-tests.
	Check that the keyboard and monitor are properly connected and powered on.
	If the power up screen is not displayed, yet the system enters console mode when you press the Return key, check that the console environment variable is set correctly. If you are using a VGA console terminal, the console variable should be set to "graphics." If you are using a serial console terminal, the console variable should be set to "serial."
	If console is set to serial, the power-up screen is routed to the COM1 serial communication port (Section 5.7) and cannot be viewed from the VGA monitor.
	Try connecting a console terminal to the COM1 serial communication port (Section 5.7). If necessary, use an MMJ-to-9-pin adapter (H8571-J). Check the baud rate setting for console terminal and system. The system baud rate setting is 9600. When using the COM1 port, you must set the console environment variable to "serial."
	If the system has a customized NVRAM file, try powering up with the Halt button set to the "in" position. The NVRAM file will not be executed when powering up with the Halt button depressed.
	For certain situations, power up using the fail-safe loader (Section 2.6) to bypass the power-up script and get to a low-level console. From the fail-safe loader console, you can edit the NVRAM file, set and examine environment variables, and initialize drivers.

Table 1–3 Diagnostic Flow for Problems Reported by the Console Program

Symptom	Action
Power-up tests do not complete.	Use power-up/diagnostic display on the operator control panel (Section 2.1) and/or console terminal (Section 2.2) to determine error.
The system powers up to the "`ash>`" prompt.	Reinstall firmware. Refer to the procedure provided with the firmware update documentation.
Console program reports error: • OCP displays failure message at power-up • Halt button LED lights during power-up • Power-up screen includes error messages • Console prompt indicates a CPU failover	Use power-up/diagnostic display on the operator control panel (Section 2.1) and/or console terminal (Section 2.2) to determine error. Use the `show fru` (Section 3.3.3) and `show error` (Section 3.3.4) commands to see if errors have been logged and to examine error information contained in serial control bus EEPROMs. Examine the console event log (enter the `cat el` command) or power-up screen (Section 2.2.2) to check for embedded error messages recorded during power-up. If power-up screen or console event log indicate problems with mass storage devices, or if storage devices are missing from the `show config` display, use the troubleshooting table (Section 2.3) to determine the problem. If power-up screen or console event log indicate problems with PCI devices, or if PCI devices are missing from the `show config` display, use the troubleshooting table (Section 2.4) to determine the problem. To troubleshoot VME problems, or if the PCI to VME daughter board is missing from the `show config` display, use the troubleshooting table (Section 2.5) to determine the problem. Run RBD tests (Section 3.1) to verify problem.

Table 1–4 Diagnostic Flow for Boot Problems

Symptom	Action
System cannot find boot device.	Check system configuration for correct device parameters (node ID, device name, and so on). • For Digital UNIX and OpenVMS, use the `show config` and `show device` (Section 5.1). Check the system configuration for correct environment variable settings. • For Digital UNIX and OpenVMS, examine the auto_action, bootdef_dev, boot_osflags, and os_type environment variables (Section 5.1.3.4). For problems booting over a network, check the ew*0_protocols or er*0_protocols environment variable settings: Systems booting from a Digital UNIX server should be set to bootp; systems booting from an OpenVMS server should be set to mop (Section 5.1.3.4).
Device does not boot.	For problems booting over a network, check the ew*0_protocols or er*0_protocols environment variable settings: Systems booting from a Digital UNIX server should be set to bootp; systems booting from an OpenVMS server should be set to mop (Section 5.1.3.4). Check that the Halt button is not set to “in” (depressed). Run device tests (Section 3.1) to check that boot device is operating.

Table 1–5 Diagnostic Flow for Errors Reported by the Operating System

Symptom	Action
System is hung or has crashed.	Examine the crash dump file.
	Refer to *OpenVMS Alpha System Dump Analyzer Utility Manual* for information on how to interpret OpenVMS crash dump files.
	Refer to the *Guide to Kernel Debugging* (AA–PS2TA–TE) for information on using the Digital UNIX Krash Utility.
	Use the `show error` command (Section 3.3.4) to examine error information contained in serial control bus EEPROMs (console environment error log).
Errors have been logged and the operating system is up.	Examine the operating system error log files to isolate the problem (Chapter 4).
	If the problem occurs intermittently, run an operating system exerciser, such as DEC VET, to stress the system.
	Refer to the *DEC Verifier and Exerciser Tool User's Guide* (AA–PTTMA–TE) for instructions on running DEC VET.

1.2 Service Tools and Utilities

This section lists the array of service tools and utilities available for acceptance testing, diagnosis, and serviceability and provides recommendations for their use.

Error Handling/Logging

Digital UNIX and OpenVMS operating systems provide recovery from errors, fault handling, and event logging. The DECevent Translation and Reporting Utility for OpenVMS and Error Report Formatter (ERF) provides bit-to-text translation of event logs for interpretation. Digital UNIX uses uerf to present the same kinds of information.

RECOMMENDED USE: Analysis of error logs is the primary method of diagnosis and fault isolation. If the system is up, or you are able to bring it up, look at this information first. Refer to Chapter 4 for information on using error logs to isolate faults.

ROM-Based Diagnostics (RBDs)

Many ROM-based diagnostics and exercisers are embedded in Digital Alpha 2100 VME systems. ROM-based diagnostics execute automatically at power-up and can be invoked in console mode using console commands.

RECOMMENDED USE: ROM-based diagnostics are the primary means of testing the console environment and diagnosing the CPU, memory, Ethernet, I/O buses, and SCSI and DSSI subsystems. Use ROM-based diagnostics in the acceptance test procedures when you install a system, add a memory module, or replace the following: CPU module, memory module, motherboard, standard I/O module, I/O bus device, or storage device. Refer to Chapter 3 for information on running ROM-based diagnostics.

Loopback Tests

Internal and external loopback tests are used to isolate a failure by testing segments of a particular control or data path. The loopback tests are a subset of the ROM-based diagnostics.

RECOMMENDED USE: Use loopback tests to isolate problems with the COM2 serial port, the parallel port, and Ethernet controllers. Refer to Chapter 3 for instructions on performing loopback tests.

Firmware Console Commands

Console commands are used to set and examine environment variables and device parameters, as well as to invoke ROM-based diagnostics and exercisers. For example, the `show memory`, `show configuration`, and `show device` commands are used to examine the configuration; the `set` (bootdef_dev, auto_action, and boot_osflags) commands are used to set environment variables.

RECOMMENDED USE: Use console commands to set and examine environment variables and device parameters and to run RBDs. Refer to Section 5.1 for information on configuration-related firmware commands and Chapter 3 for information on running RBDs.

Operating System Exercisers (DEC VET)

The Digital Verifier and Exerciser Tool (DEC VET) is supported by the Digital UNIX and OpenVMS operating systems. DEC VET performs exerciser-oriented maintenance testing of both the hardware and operating system.

RECOMMENDED USE: Use DEC VET as part of acceptance testing to ensure that the CPU, memory, disk, tape, file system, and network are interacting properly. Also use DEC VET to stress test the user's environment and configuration by simulating system operation under heavy loads to diagnose intermittent system failures.

Crash Dumps

For fatal errors, such as fatal bugchecks, Digital UNIX and OpenVMS operating systems will save the memory contents to a crash dump file.

RECOMMENDED USE: Crash dump files can be used to determine why the system crashed. To save a crash dump file for analysis, you need to know proper system settings. Refer to the *OpenVMS Alpha System Dump Analyzer Utility Manual* or the *Guide to Kernel Debugging* (AA–PS2TA–TE) for Digital UNIX.

Recommended System Installation

The recommended system installation includes:

1. Hardware installation and acceptance testing. Acceptance testing includes running the `test` command.
2. Software installation and acceptance testing. For example, using OpenVMS Factory Installed Software (FIS) and then acceptance testing with DEC VET.

1.3 Information Services

Several information resources are available, including online information for servicers and customers, computer-based training, and maintenance documentation database services. A brief description of some of these resources follows.

Training

Computer Based Training (CBT) and lecture lab courses are available from the Digital training center:

- AlphaServer 2100 Installation and Troubleshooting: EY–M915E
- Alpha Concepts
- RAID Concepts: EY–N935E
- SCSI Concepts and Troubleshooting: EY–P841E, EY–N838E

Digital Assisted Services

Digital Assisted Services (DAS) offers products, services, and programs to customers who participate in the maintenance of Digital computer equipment. Components of Digital Assisted Services include:

- Spare parts and kits
- Diagnostics and service information/documentation
- Tools and test equipment
- Parts repair services, including Field Change Orders

2

Power-Up Diagnostics and Displays

This chapter provides information on how to interpret the power-up/diagnostic display on the operator control panel and console screen. In addition, a description of the power-up and firmware power-up diagnostics is provided as a resource to aid in troubleshooting.

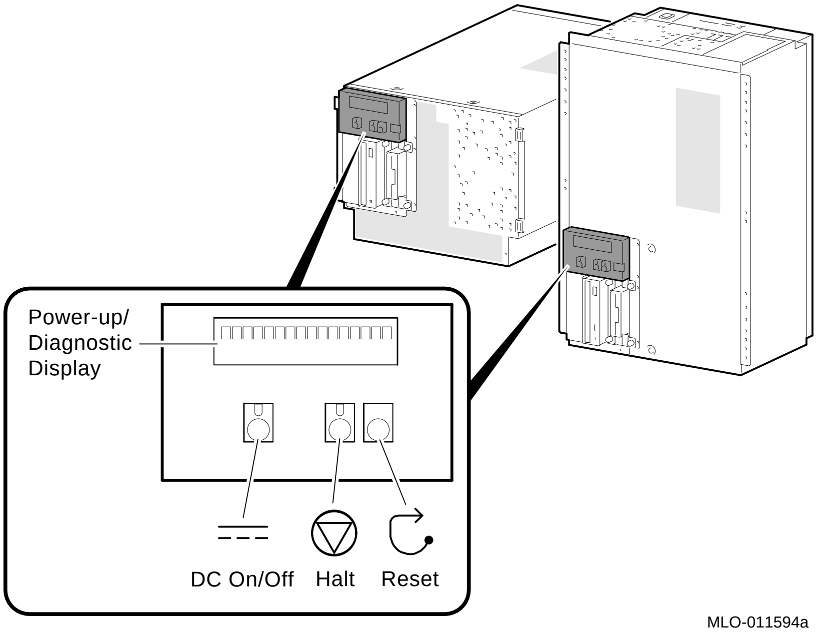
- Section 2.1 describes how to interpret the power-up/diagnostic display on the operator control panel.
- Section 2.2 describes how to interpret the power-up screen.
- Section 2.3 describes how to troubleshoot mass-storage problems indicated at power-up or storage devices missing from the `show config` display.
- Section 2.4 describes how to troubleshoot PCI bus problems or PCI devices missing from the `show config` display.
- Section 2.5 describes how to troubleshoot VME bus problems.
- Section 2.6 describes the use of the fail-safe loader.
- Section 2.7 describes how to interpret system LEDs.
- Section 2.8 describes the power-up sequence.
- Section 2.9 describes power-on self-tests.

2.1 Interpreting the Power-Up Display

The power-up/diagnostic display on the operator control panel (OCP) (Figure 2–1) displays the progress and results of self-tests during power-up.

The OCP power-up display is the primary diagnostic tool for troubleshooting “No Access to Console Mode” problems.

Figure 2–1 Operator Control Panel Power-Up/Diagnostic Display



Tables 2–1 and 2–2 contain information on interpreting the diagnostic display.

Table 2–1 Interpreting OCP Power-Up Display

Message	Meaning
TEST	Displayed while system performs diagnostic tests and exercisers. The type of module under test, its slot number, and the currently executing test number are also displayed.
NO MEM INSTALLED	Displayed if you power up with no memory installed.
FAIL module_type	If an error is detected in the CPU, memory, or I/O, a failure message is displayed and the Halt button LED lights for a few seconds. The error is logged to the appropriate module via the serial control bus. In nearly all cases, the power-up tests continue. The module_type and slot number for the field replaceable unit (FRU) that failed, along with the test number (Table 2–2) that detected the error are also displayed. Module types and slot numbers: CPU_nn — CPU module (0–3) MEM_nn — Memory module (0–3) I/O_0 — Standard I/O module
CPU STATUS	Summary of CPU testing. The status of each CPU, starting with CPU0 is displayed: "P" — CPU passed "F" — CPU failed "--" — CPU not present
STARTING CPU #	The console is starting the primary CPU.
TEST MEM BANK #	The console is testing memory.
PROBE I/O SUBSYS	The console is checking the PCI and EISA bridges.
SYSTEM RESET	The Reset button has been pressed.
Model x/xxx	When system is under operating system control, the CPU variant (*x*) and the approximate CPU speed (*xxx*) are displayed unless you supply your own text using the ocp_ text environment variable.

Table 2–2 Serial ROM Power-Up Test Description and Field Replaceable Units (FRUs)

Test Number	Description	Likely FRU
95	SROM unloaded, sync byte sent to the DECchip 21064 processor	CPU
91	Sync byte received from the DECchip 21064 processor	CPU
8d	First backup cache initialization	CPU
89	Backup cache data line test	CPU
85	Backup cache graycode test	CPU
81	DECchip 21064 processor ECC generation test	CPU
7d	Backup cache tag store test	CPU
79	I/O tests: System bus, PCI bus, EISA bus	CPU, standard I/O, or motherboard[1]
75	Second backup cache initialization	CPU
0c	End of initial test sequence (CPU and all buses good)	CPU
18	Memory 0	MEM
16	Memory 1	MEM
14	Memory 2	MEM
12	Memory 3	MEM
07-CPU#	End of memory test (32 MB)	MEM
06	Start ESC configuration	I/O_0
05	End of ESC config/start FEPROM unload	I/O_0
04	End of FEPROM unload/start checking	I/O_0
03	End of checking, jump to unloaded console	I/O_0

(1) Use the `show error cpu` command to isolate the failing FRU. If an error log indicates that the CPU failed test number 7, the CPU module is faulty.

If no error is logged for test number 7, the standard I/O is the likely module at fault. If replacing the standard I/O module does not solve the problem, the system bus motherboard is probably faulty.

2.2 Power-Up Screen

During power-up self-tests the test status and result are displayed on the console terminal. Information similar to the following example should be displayed on the screen.

```
starting console on CPU 0
initialized idel PCB
initializing semaphores
initializing heap
Initial heap 1c0c0
memory low limit = 100000
heap = 1c0c0 13fe0
initializing driver structures
initializing idel process PID
XDELTA not enabled
initializing file system
initializing timer data structures
lowering IPL
counted 92780731 cycles in 500 ticks
CPU 0 speed is 5.26 ns (190 MHz)
access NVRAM
entering idle loop
Starting Memory Diagnostics
Testing CSIC on Memory Module 0
Testing all memory banks in parallel
Testing Memory bank 0
Testing Memory bank 1
Configuring Memory Modules
Configuring memory size = 4000000
Memory Diagnostics completed
probing hose 0, PCI
bus 0, slot 0 -- ewa -- DECchip 21040-AA
bus 0, slot 1 -- pka -- NCR 53C810
bus 0, slot 2 --     -- Intel 82375EB
bus 0, slot 6 --     -- DECchip 21040-AA
probing PCI-to-PCI bridge, bus 1
probing hose 1, EISA
probing hose 2, PCI
initializing keyboard
Memory Testing and Configuration Status
Module   Size    Base Addr   Intlv Mode  Intlv Unit  Status
------   -----   ---------   ----------  ----------  ------
  1       64MB   00000000      1-Way         0       Passed
Total Bad Pages 0
Testing the System
Testing the Disks (read only)
Change to Internal loopback
Testing the Network
Change to Normal Operating Mode
environment variable mopV3_boot created
AlphaServer 2100 Console V3.8-49, built on Nov 7 1994 at 12:22:36
P00>>>
```

Note

To stop the screen display from scrolling, press Ctrl/S. To resume scrolling, press Ctrl/Q.

Digital UNIX or OpenVMS Systems

Digital UNIX and OpenVMS are supported by the SRM firmware (see Section 5.1.1). The SRM console prompt is shown below as: `P00>>>`

Note

For systems with multiple CPUs, if CPU0 failed during power-up tests, or has an error logged to its EEPROM, the system will "failover" to another CPU. The number of the CPU serving as the primary CPU is displayed in the SRM prompt. For example, `P01>>>` or `P02>>>`, and so on.

2.2.1 Multiprocessor Failover

Digital Alpha 2100 VME systems support multiprocessor failover, which allows the system to power up and boot the operating system even if only one CPU is working.

During power-up or system reset, the serial ROM tests check for a good CPU, starting with CPU0, to serve as the primary CPU. The primary CPU is the only CPU that tests memory and reads the flash ROM code. If a CPU fails serial ROM tests, or if the CPU has an error logged to its serial control bus EEPROM, that CPU is disabled. The lowest-numbered passing CPU serves as the primary CPU. If all CPU modules fail their power-up diagnostics, then CPU0 will serve as the primary CPU.

If any of the CPUs fail during power-up, the Halt button LED on the operator control panel lights for a few seconds and the CPU status message on the power-up/diagnostic display indicates which CPU failed (Table 2–1).

The following firmware commands can also be used to determine if a CPU failed power-up tests.

- `show fru` (Chapter 3)
- `show error` (Chapter 3)
- `show config` (Chapter 5)

Note

The CPU number of the CPU serving as the primary CPU is displayed in the SRM prompt. For example, `P01>>>` or `P02>>>`, and so on.

2.2.2 Console Event Log

Digital Alpha 2100 VME systems maintain a console event log consisting of status messages received during power-on self-tests. If problems occur during power-up, standard error messages indicated by asterisks (***) may be embedded in the console event log. To display a console event log, use the `cat el` command.

Note

To stop the screen display from scrolling, press Ctrl/S. To resume scrolling, press Ctrl/Q.

You can also use the `more el` command to display the console event log one screen at a time.

The following examples show abbreviated console event logs that contain standard error messages.

❶ The first message indicates a problem with the keyboard.

❷ The second indicates that the Ethernet loopback test failed (possibly the result of a missing terminator or disconnection from a live network).

```
P00>>> cat el
starting console on CPU 0
initialized idle PCB
initializing semaphores
initializing heap
   .
   .
   .
CPU 0 speed is 5.26 ns (190MHz)
access NVRAM
entering idle loop
Starting Memory Diagnostics
   .
   .
   .
initializing keyboard
*** Keyboard not plugged in...  ❶

** keyboard error **
   .
   .
   .
Change mode to Internal loopback.

*** Error (ewa0), Mop loop message timed out from: 08-00-2b-3d-63-10  ❷

*** List index: 0 received count: 0 expected count 1
   .
   .
   .
Change to Normal Operating Mode.
environment variable mopv3_boot created
P00>>>
```

2.3 Mass Storage Problems Indicated at Power-Up

Mass storage failures at power-up are usually indicated by read fail messages. Other problems are indicated by storage devices missing from the `show config` display.

- Table 2–3 provides information for troubleshooting fixed-media mass storage problems indicated at power-up or storage devices missing from the `show config` display.
- Table 2–4 provides information for troubleshooting removable-media storage problems indicated at power-up or storage devices missing from the `show config` display.

Use the information in Tables 2–3 and 2–4 to diagnose the likely cause of the problem.

Table 2–3 Fixed-Media Mass Storage Problems

Problem	Symptom	Corrective Action
Drive failure	Fault LED for drive is on (steady).	Replace drive.
Duplicate SCSI IDs (when removable-media bus is extended to StorageWorks shelf)	Drives with duplicate SCSI IDs are missing from the `show config` display.	Correct removable-media SCSI IDs.
SCSI ID set to 7 (reserved for host ID)	Valid drives are missing from the `show config` display. One drive may appear seven times on the configuration screen display.	Correct SCSI IDs.
Duplicate host IDs on a shared bus	Valid drives are missing from the `show config` display. One drive may appear seven times on the configuration screen display.	Change host ID through the pk*0_host_id environment variable (`set pk*0_host_id`).
Extra terminator	Devices produce errors or device IDs are dropped.	Check that bus is terminated only at beginning and end. Remove unnecessary terminators.

(continued on next page)

Table 2–3 (Cont.) Fixed-Media Mass Storage Problems

Problem	Symptom	Corrective Action
Standard I/O module failure (if native SCSI bus is extended to fixed disk drives) or PCI or VME storage adapter option failure	Problems persist after eliminating the above problem sources.	Replace storage adapter module or standard I/O.

Table 2–4 Removable-Media Mass Storage Problems

Problem	Symptom	Corrective Action
Drive failure	Fault LED for drive is on (steady).	Replace drive.
Duplicate SCSI IDs	Drives with duplicate SCSI IDs are missing from the `show config` display.	Correct SCSI IDs.
SCSI ID set to 7 (reserved for host ID)	Valid drives are missing from the `show config` display. One drive may appear seven times on the `show config` display.	Correct SCSI IDs.
Duplicate host IDs on a shared bus	Valid drives are missing from the `show config` display. One drive may appear seven times on the configuration screen display.	Change host ID through the pk*0_host_id environment variable (`set pk*0_host_id`).
Missing or loose cables	Activity LEDs do not come on. Drive missing from the `show config` display.	Remove device and inspect cable connections.
Terminator missing	Read/write errors in console event log; storage adapter port may fail.	Attach terminators as needed: Internal SCSI terminator (12-41296-01) or external SCSI terminator (12-37004-04).
Standard I/O module failure	Problems persist after eliminating the previous problem sources.	Replace standard I/O module.

2.4 PCI Bus Problems

PCI bus failures are usually indicated by the inability of the system to see the device. Table 2–5 provides information for troubleshooting PCI bus problems. Use the table to diagnose the likely cause of the problem.

Note

1. Some PCI devices do not implement PCI parity, and some have a parity-generating scheme in which parity is sometimes incorrect or is not compliant with the PCI Specification. In such cases, the device functions properly as long as parity is not checked. The pci_parity environment variable for for the SRM console, or the DISABLEPCIPARITY CHECKING for the ARC console, allow you to turn off parity checking so that false PCI parity errors do not result in machine check errors.

   When you disable PCI parity, no parity checking is implemented for any PCI device, even those devices that produce correct, compliant parity.

2. Some PCI devices that are not compliant with the PCI specification may not function properly, particularly in slot PCI1.

Table 2–5 PCI Troubleshooting

Step	Action
1	Confirm that the PCI module and any cabling are properly seated.
2	Run ROM-based diagnostics for the type of option: • Storage adapter—Run `exer_read` to exercise the storage devices off the PCI controller option (Section 3.3.6). • Ethernet adapter—Run `nettest` to exercise an Ethernet adapter (Section 3.3.9).
3	Check for a bad slot by moving the last installed controller to a different slot.
4	Call option manufacturer or support for help.

2.5 VME Bus Problems

Table 2–6 provides information for troubleshooting VME bus problems. Use the table to diagnose the likely cause of the problem.

Table 2–6 VME Troubleshooting

Step	Action
1	Confirm that the VME module and any cabling are properly seated.
2	Enter the `show config` command to make sure the PCI to VME daughter board is recognized: Look for the DECchip 7407 in slot 0 under Hose 0, Bus 1, PCI. If the `show config` display does not not include the DECchip 7407, than the PCI to VME daughter board may be defective.
3	Check that the VME jumpers are set correctly according to the configuration (Section 5.5.)
4	Check for a bad slot by moving the last installed VME option to a different slot.
5	Call option manufacturer or support for help.

2.6 Fail-Safe Loader

The fail-safe loader (FSL) allows you to power up without initializing drivers or running power-up diagnostics.

Note

The fail-safe loader should be used only when a failure at power-up prohibits you from getting to the console program. You cannot boot an operating system from the fail-safe loader.

If a checksum error is detected when the SRM console is loading at power-up, the fail-safe loader is automatically loaded into memory and the system displays the FSL prompt `ash>`. If the system automatically powers up to the `ash>` prompt, reinstall console firmware according to the instructions provided with the firmware.

Whenever the fail-safe loader console is activated, the power-up/diagnostic display on the operator control panel displays a `FAIL I/O_00` message.

The FSL permits you to get to a console, with limited functionality, when one of the following is the cause of a problem getting to the console program under normal power-up:

- A power failure or accidental power down during a firmware upgrade

- An error in the nonvolatile NVRAM file
- An incorrect environment variable setting
- A driver error

Note

The FSL program, indicated by the `ash>` prompt, has limited functionality. A simple shell is indicated by the letters "ash" contained in the console prompt.

2.6.1 Fail-Safe Loader Functions

From the FSL program, you can:

- Load new console firmware
- Edit the nvram file (using the `edit` command)
- Assign a correct value to an environment variable (using the `show` and `set` commands)
- Start individual drivers using the `init -driver ew` command to start the MOP driver or `init -driver dv` to start the floppy driver. The `init -driver 6` command in FSL mode starts all available drivers.

Note

The nonvolatile file, NVRAM, is shipped from the factory with no contents. The customer can use the `edit` command to create a customized script or command file that is executed as the last step of every power-up.

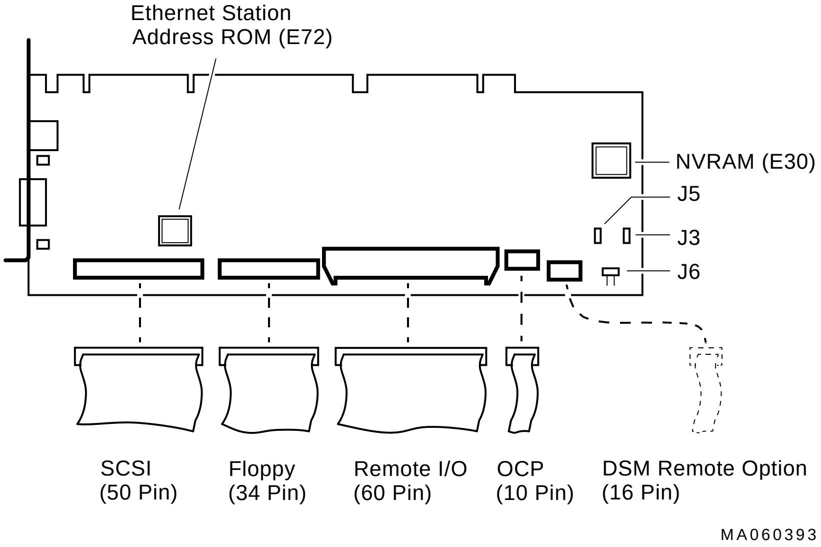
2.6.2 Activating the Fail-Safe Loader

To activate the FSL:

1. Install jumper J6 on the standard I/O module (Figure 2–2). The jumper is stored on one of the pins of the J6 jumper.
2. Turn on the system.
3. Use the FSL program (`ash>`) to make corrections, edit the NVRAM file, set environment variables, or initialize phase 6 drivers.

4. When you have finished, power down and remove the FSL jumper.

Figure 2–2 Fail-Safe Loader Jumper (J6) on the Standard I/O Module



J3–Power supply mode: Digital Alpha VME 2100 systems use the full power mode setting (jumper not installed).
J5–Program voltage: Internal use only.
J6–Fail-Safe: When installed, selects the fail-safe loader firmware.

2.7 Interpreting System LEDs

This section describes the function of system LEDs and what action to take when a failure is indicated. The system LEDs are used primarily to troubleshoot power problems and problems with boot devices. There are four types of system LEDs:

- Halt button LED at power-up
- Storage device LEDs
- I/O panel LEDs

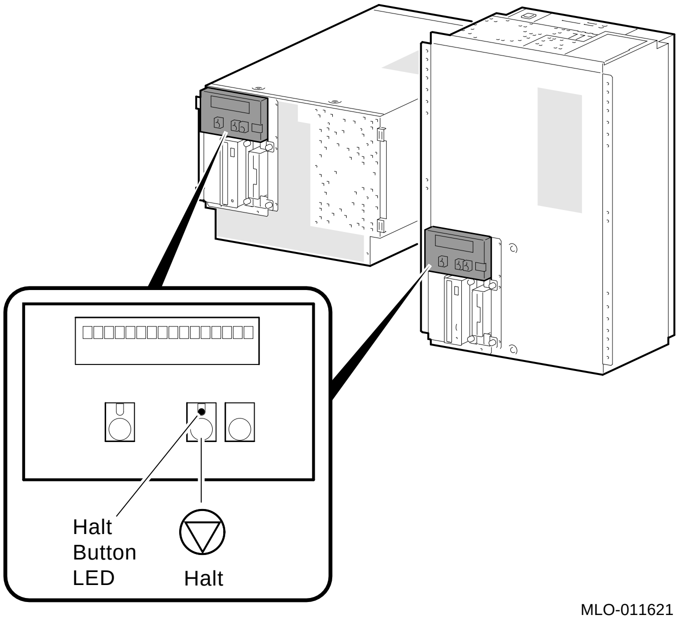
2.7.1 Halt Button LED (At Power Up)

During power-up, the console firmware checks for errors logged through the serial control bus. If an error is detected, the Halt button LED on the operator control panel lights.

If the Halt button LED comes on during power-up, use the `show fru` and `show error` commands (Chapter 3) to see what errors have been logged and to examine error information contained in serial control bus EEPROMs.

Figure 2–3 shows the location of the Halt button LED.

Figure 2–3 Halt Button

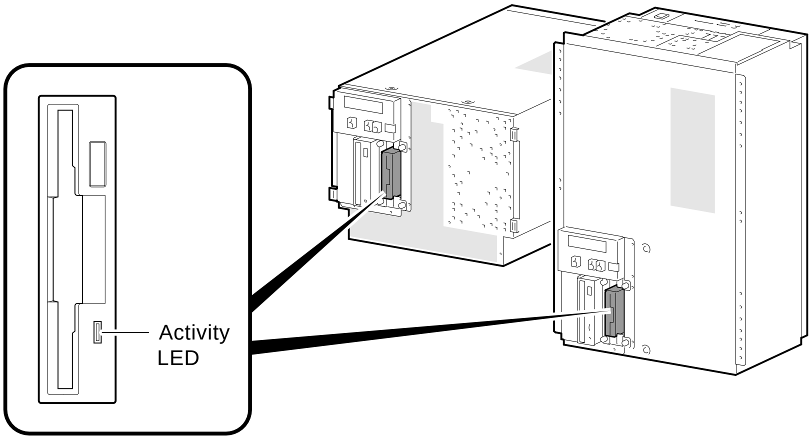


2.7.2 Storage Device LEDs

Storage device LEDs indicate the status of the device.

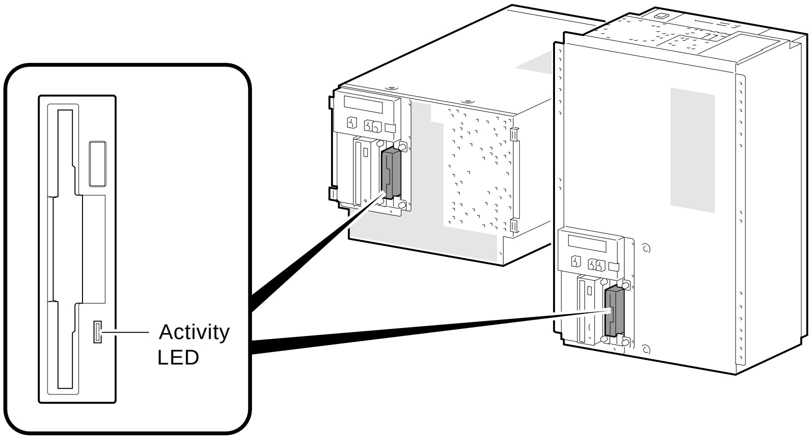
- Figure 2–4 shows the Activity LED for the floppy drive. This LED is on when the drive is in use.
- Figure 2–5 shows the Activity LED for the CD–ROM drive. This LED is on when the drive is in use.

Figure 2–4 Floppy Drive Activity LED



MLO-011633

Figure 2–5 CD–ROM Drive Activity LED



MLO-011633

For information on other storage devices, refer to the documentation provided by the manufacturer or vendor.

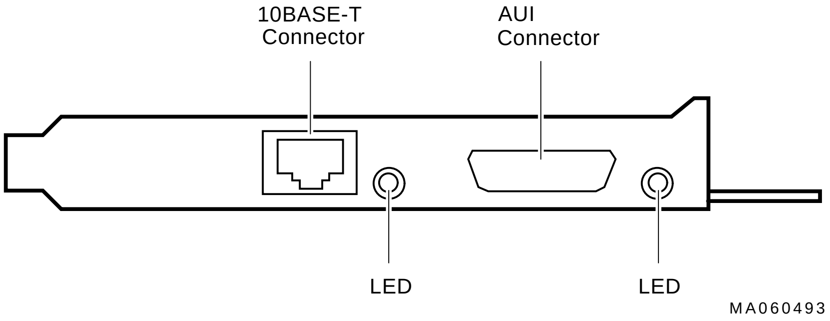
2.7.3 Standard I/O Panel LEDs

The standard I/O panel LEDs (Figure 2–6) indicate which Ethernet port is currently selected, 10BASE-T or AUI.

Use the `ew*0_mode` environment variable to select the default Ethernet device type:

- aui — Sets the default Ethernet device to AUI.
- twisted — Sets the default Ethernet device type to 10BASE-T (twisted-pair).
- auto — Reads the device connected to the Ethernet port and sets the default to the appropriate Ethernet device type.

Figure 2–6 Standard I/O Panel LEDs



2.8 Power-Up Sequence

During the Digital Alpha 2100 VME power-up sequence, the power supply is stabilized and the system is initialized and tested via the firmware power-on self-tests.

The power-up sequence includes the following:

- Power supply power-up:
  - AC power-up
  - DC power-up
- Two sets of power-on diagnostics:
  - Serial ROM diagnostics
  - Console firmware-based diagnostics

2.8.1 AC Power-Up Sequence

The following power-up sequence occurs when AC power is applied to the system (system is plugged in) or when electricity is restored after a power outage:

1. The front end of the power supply begins operation and energizes.
2. The power supply then waits for the DC power to be enabled.

2.8.2 DC Power-Up Sequence

DC power is applied to the system with the DC On/Off button on the operator control panel.

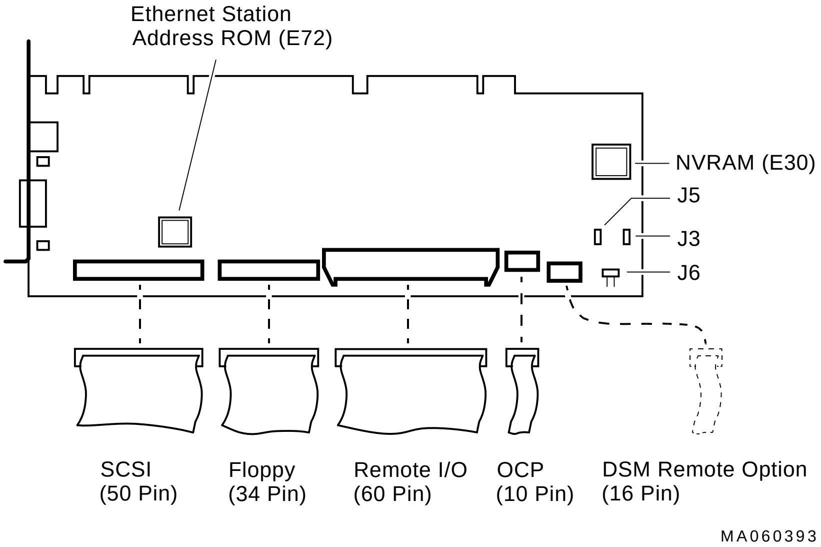
A summary of the DC power-up sequence is provided as follows:

1. When the DC On/Off button is pressed, the power supply outputs are enabled.
2. 12V, 5V, 3.3V, and -12V outputs are energized and stabilized. If the outputs do not come into regulation, the power-up is aborted and the power supply enters the latching-shutdown mode.
3. With DC voltages stabilized, the power supply delivers a POK_H signal to the standard I/O module and motherboard.

> **Note**
>
> You should hear the system cooling fans spin up at this point in the power-up sequence.

4. Firmware power-up diagnostics begin.

Figure 2–7 Power Supply Mode Jumper (J3) on the Standard I/O Module



J3–Power supply mode: Digital Alpha VME 2100 use the full power mode setting (jumper not installed).
J5–Program voltage: Internal use only.
J6–Fail-Safe: When installed, selects the fail-safe loader firmware.

2.9 Firmware Power-Up Diagnostics

After successful completion of AC and DC power-up sequences, the processor performs its power-up diagnostics. These tests verify system operation, load the system console, and test the core system (CPU, memory, standard I/O module, and motherboard), including all boot path devices. These tests are performed as two distinct sets of diagnostics:

1. Serial ROM diagnostics—These tests are loaded from the serial ROM located on the CPU module into the CPU's instruction cache (I-cache). They check the basic functionality of the system and load the console code from the FEPROM on the standard I/O module into system memory.

   Failures during these tests are indicated by the power-up/diagnostic display on the operator control panel. Diagnostic test and exerciser failures are also logged in EEPROM as test-directed diagnostic (TDD) error logs via the serial control bus for CPU, memory, and standard I/O modules.

2. Console firmware-based diagnostics—These tests are executed by the console code. They test the core system, including all boot path devices.

Failures during these tests are reported to the console terminal through the power-up screen or console event log. Diagnostic test and exerciser failures are also logged in EEPROM as TDD or symptom-directed diagnostic (SDD) error logs through the serial control bus for CPU, memory, and standard I/O modules.

2.9.1 Serial ROM Diagnostics

The serial ROM diagnostics are loaded into the CPU's instruction cache from the serial ROM on the CPU module. They test the system in the following order:

1. Test the CPU and backup cache located on the CPU module. If the backup cache fails testing, a CPU failure is indicated on the power-up/diagnostic display on the operator control panel (OCP), the error is logged to the serial control bus EEPROM, and remaining backup cache tests are completed.
2. Test the CPU module's system bus interface.
3. Test the system bus to PCI bus bridge and PCI bus to EISA bus bridge. If the PCI bridge fails or EISA bridge fails, a standard I/O failure is indicated on the power-up/diagnostic display on the OCP. The power-up tests continue despite these errors.
4. CPUs determine which CPU will serve as the primary CPU. Each CPU reads error log information from every CPU EEPROM. The lowest-numbered passing CPU is selected as the primary CPU in a process called multiprocessor failover (Section 2.2.1). If all CPUs fail power-up diagnostics, then CPU0 is selected as the primary CPU. The primary CPU then takes control and completes the remaining steps.
5. Locate the largest memory module in the system and test the first 32 MB of memory on the module. Only the first 32 MB of memory are tested. If there is more than one memory module of the same size, the lowest-numbered memory module (one closest to the CPU) is tested first.

   If the memory test fails, the next largest memory module in the system is tested. Testing continues until a good memory module is found. If a good memory module is not found, a memory failure is indicated on the power-up /diagnostic display on the OCP, and the power-up tests are terminated.
6. Check the access to the FEPROMs on the standard I/O module.
7. The SRM console program is loaded into memory from the FEPROM on the standard I/O module. A checksum test is executed for the console image. If the checksum test fails, the fail-safe loader (FSL) is automatically loaded into memory and the system displays the FSL prompt, `ash>`.

   If the checksum test passes, control is passed to the console code and the console firmware-based diagnostics are run.

While the console is being loaded into memory, CPUs with errors logged are disabled (if not the primary CPU). Working CPUs spin on mailbox (they continuously read the mailbox address).

2.9.2 Console Firmware-Based Diagnostics

Console firmware-based tests are executed once control is passed to the console code in memory. They check the system in the following order:

1. Perform a complete check of system memory. If a system has more than one memory module, the modules are checked in parallel.
2. Set memory interleave to maximize interleave factor across as many memory modules as possible (one, two, or four-way interleaving). During this time the console firmware is moved into backup cache on the primary CPU module. After memory interleave is set, the console firmware is moved back into memory.

   Steps 3–6 may be completed in parallel.
3. Start the I/O drivers for mass storage devices and tapes. At this time a complete functional check of the machine is made. After the I/O drivers are started, the console program continuously polls the bus for devices approximately every 20 or 30 seconds.
4. Check that EISA configuration information is present in NVRAM for EISA devices on the standard I/O module.
5. Run exercisers on the drives currently seen by the system.

   Note

   This step does not ensure that all disks in the system will be tested or that any device drivers will be completely tested. Spin-up time varies for different drives, so not all disks may be online at this point in the power-up. To ensure complete testing of disk devices, use the `test` command.

6. If the Halt button is set to “in” (depressed), the customized NVRAM script (if the customer has created one) is not executed.
7. Enter console mode or boot the operating system. This action is determined by the Halt button setting or `auto_action` environment variable.

3
Running System Diagnostics

This chapter provides information on how to run system diagnostics.

- Section 3.1 describes how to run ROM-based diagnostics, including error reporting utilities and loopback tests.
- Section 3.2 provides a summary of diagnostic and related commands.
- Section 3.3 provides detailed information for diagnostic and related commands.
- Section 3.4 describes acceptance testing and initialization procedures.
- Section 3.5 describes the DEC VET operating system exerciser.

3.1 Running ROM-Based Diagnostics

ROM-based diagnostics (RBDs), which are part of the console firmware that is loaded from the FEPROM on the standard I/O module, offer many powerful diagnostic utilities, including the ability to examine error logs from the console environment and run system- or device-specific exercisers.

Digital Alpha 2100 VME RBDs rely on exerciser modules, rather than functional tests, to isolate errors. The exercisers are designed to run concurrently, providing a maximum bus interaction between the console drivers and the target devices.

The multitasking ability of the console firmware allows you to run diagnostics in the background (using the background operator “&” at the end of the command). You run RBDs by using console commands.

Note

> ROM-based diagnostics, including the `test` command, are run from the SRM console (firmware used by OpenVMS and Digital UNIX).

RBD console commands do not log errors to the serial control bus EEPROMs. Errors are reported to the console terminal and/or the console event log.

3.2 Command Summary

Table 3–1 provides a summary of the diagnostic and related commands.

Table 3–1 Summary of Diagnostic and Related Commands

Command	Function	Reference
Acceptance Testing		
test	Quickly tests the core system. The test command is the primary diagnostic for acceptance testing and console environment diagnosis.	Section 3.3.1
Error Reporting		
clear_error	Clears error information logged through the serial control bus. The show error command displays errors logged to the serial control bus EEROMs.	Section 3.3.5
show error	Reports core system errors captured by test-directed diagnostics (TDD) through the RBDs and symptom-directed diagnostics (SDD) through the operating system.	Section 3.3.4
show fru	Reports system bus module identification numbers and summary error information.	Section 3.3.3

(continued on next page)

Table 3–1 (Cont.) Summary of Diagnostic and Related Commands

Command	Function	Reference
Extended Testing/Troubleshooting		
exer_read	Tests a disk by performing random reads on the specified device.	Section 3.3.6
memexer	Exercises memory by running a specified number of memory tests. The tests are run in the background.	Section 3.3.7
memexer_mp	Tests memory in a multiprocessor system by running a specified number of memory exerciser sets. The tests are run in the background.	Section 3.3.8
net -ic	Initializes the MOP counters for the specified Ethernet port.	Section 3.3.11
net -s	Displays the MOP counters for the specified Ethernet port.	Section 3.3.10
nettest	Runs external loopback tests for specified PCI-based Ethernet ports.	Section 3.3.9
sys_exer	Exercises core system. Runs tests concurrently.	Section 3.3.2
Loopback Testing		
test lb	Conducts loopback tests for COM2 and the parallel port in addition to quick core system tests.	Section 3.3.1
sys_exer lb	Conducts loopback tests for COM2 and the parallel port in addition to core system tests.	Section 3.3.2
nettest	Runs external or internal loopback tests for specified PCI-based Ethernet ports.	Section 3.3.9
Diagnostic-Related Commands		
kill	Terminates a specified process.	Section 3.3.12
kill_diags	Terminates all currently executing diagnostics.	Section 3.3.12
show_status	Reports the status of currently executing test /exercisers.	Section 3.3.13

3.3 Command Reference

This section provides detailed information on the diagnostics commands and related commands.

3.3.1 test

The `test` command runs firmware diagnostics for the entire core system. The tests are run sequentially and the status of each subsystem test is displayed to the console terminal as the tests progress. If a particular device is not available to test, a message is displayed.

Note

By default, no write tests are performed on disk and tape drives. Media must be installed to test the floppy drive and tape drives.

The `test` script tests devices in the following order:

1. Memory tests (one pass)

   Note

   Certain memory errors that are reported by the OCP may not be reported by the ROM-based diagnostics. Always check the power-up/diagnostic display before running diagnostic commands.

2. Read-only tests: DK* disks, DR* disks, DU* disks, MK* tapes, DV* floppy
3. Console loopback tests if lb argument is specified: COM2 serial port and parallel port. Loopback connectors should be installed on the COM2 and parallel port for these tests.
4. Network external loopback tests for EWA0—This test requires that the Ethernet port be terminated or connected to a live network; otherwise, the test will fail.

Synopsis:

test [lb]

Arguments:

[lb] The loopback option includes console loopback tests for the COM2 serial port and the parallel port during the test sequence.

Examples:

The system is tested and the tests complete successfully.

```
P00>>> test
 Testing the Memory
Testing the DK* Disks(read only)
dkb600.6.0.2.1 has no media present or is disabled via the RUN/STOP switch
file open failed for dkb600.6.0.2.1
No DR* Disks available for testing
Testing the MK* Tapes(read only)
Testing the DV* Floppy Disks(read only)
file open failed for dva0.0.0.0.1
Testing the VGA(Alphanumeric Mode only)
Testing the EW* Network
P00>>>
```

The system is tested and the system reports an error message. No network server responded to a loopback message. Ethernet connectivity on this system should be checked.

```
P00>>> test
Testing the Memory
Testing the DK* Disks(read only)
No DR* Disks available for testing
Testing the MK* Tapes(read only)
Testing the DV* Floppy Disks(read only)
Testing the VGA(Alphanumeric Mode only)
Testing the EW* Network

*** Error (ewa0), Mop loop message timed out from: 08-00-2b-3b-42-fd

*** List index: 7 received count: 0 expected count 2

P00>>>
```

3.3.2 sys_exer

The `sys_exer` command runs firmware diagnostics for the entire core system. The same tests that are run using the `test` command are run with `sys_exer`, only these tests are run concurrently and in the background. Nothing is displayed unless an error occurs.

Note

Some processes started using `sys_exer` are not stopped using the `kill` and `kill_diags` commands. Use the `init` command to terminate all sys_exer processes.

Because the sys_exer tests are run concurrently and indefinitely (until you stop them with the `init` command), they are useful in flushing out intermittent hardware problems.

Note

By default, no write tests are performed on disk and tape drives. Media must be installed to test the floppy drive and tape drives.

Certain memory errors that are reported by the OCP may not be reported by the ROM-based diagnostics. Always check the power-up/diagnostic display before running diagnostic commands.

Synopsis:

sys_exer [lb]

Arguments:

[lb]	The loopback option includes console loopback tests for the COM2 serial port and the parallel port during the test sequence.

Examples:

```
P00>>> sys_exer

Exercising the Memory
Exercising the DK* Disks(read only)
Exercising the MK* Tapes(read only)
Exercising the Floppy(read only)
Exercising the VGA(Alphanumeric Mode only)
Exercising the EWA0 network

Type init in order to boot the operating system

P00>>> show_status
 ID       Program      Device       Pass  Hard/Soft Bytes Written  Bytes Read
-------- ------------ ------------ ------ --------- ------------- -------------
00000001         idle system            0    0    0             0             0
0000006f      memtest memory            1    0    0      35651584      35651584
00000070      memtest memory            1    0    0      35651584      35651584
00000077      memtest memory            1    0    0      37748736      37748736
0000007e     exer_kid dka0.0.0.1.0      0    0    0             0         69120
0000007f     exer_kid dka600.6.0.1      0    0    0             0         66560
00000093     exer_kid dva0.0.0.0.1      0    0    0             0             0
000000d5      nettest ewa0.0.0.0.0     13    0    0        308672        308672
P00>>> init
```

3.3.3 show fru

The `show fru` command reports FRU and error information for the following FRUs based on the serial control bus EEPROM data:

- CPU modules
- Memory modules
- I/O modules

For each of the above FRUs, the slot position, option, part, revision, and serial numbers, as well as any reported symptom-directed diagnostics (SDD) and test-directed diagnostics (TDD) event logs are displayed.

In addition, installed PCI modules are displayed with their respective slot numbers.

Synopsis:

show fru ([target [target . . .]])

Arguments:

[target]	CPU{0,1,2,3}, mem{0,1,2,3}, io.

Example:

```
P00>>> show fru
  ❶     ❷       ❸                ❹         ❺               ❻
                                 Rev                    Events logged
 Slot   Option  Part#           Hw Sw   Serial#         SDD     TDD
   0    IO      B2110-AA        H2 0    KA427P0593      00      00
   2    CPU0    B2020-AA        C2 9    KA426C0457      00      00
   4    MEM0    B2021-BA        B1 0    AY43314429      00      00

 Slot   Option                  Hose 0, Bus 0, PCI
   0    DECchip 21040-AA
   1    NCR 53C810
   2    Intel 82375EB
   6    DECchip 21050-AA

 Slot   Option                  Hose 1, Bus 0, EISA

 Slot   Option                  Hose 2, Bus 0, PCI
P00>>>
```

❶ System bus slot number for FRU (slots 0–7 top to bottom)

Slot 0: Standard I/O module (dedicated PCI card cage slot)
Slot 1–3, 5: CPU modules
Slot 4–7: Memory modules

❷ Option name (I/O, CPU#, or MEM#)

❸ Part number of option

❹ Revision numbers (hardware and firmware)

❺ Serial number

❻ Events logged:

Numbers other than “00” indicate that errors have been logged.

- SDD: Number of symptom-directed diagnostic events logged by the serial ROM diagnostics at power up.
- TDD: Number of test-directed diagnostic events logged by the firmware diagnostics at power up.

3.3.4 show error

The `show error` command reports error information based on the serial control bus EEPROM data. Both the operating system and the ROM-based diagnostics log errors to the serial control bus EEPROMs. This functionality provides the ability to generate an error log from the console environment.

A closely related command, `show fru` (Section 3.3.3), reports FRU and error information for FRUs.

Synopsis:

show error ([target [target . . .]])

Arguments:

[target]	CPU{0,1,2,3}, mem{0,1,2,3}, and io.

Memory Errors

Note

Certain memory errors that are reported by the OCP may not be reported by the ROM-based diagnostics. Always check the power-up/diagnostic display before running diagnostic commands.

Correctible errors are indicated by event type 00. If five or more correctible errors are logged for the same memory module, the specified module should be replaced.

For all unncorrectible errors, indicated by event types 01 and 10, you should replace the memory module.

Only two bad memory data bits at one time are captured by the system diagnostics.

Memory Error Example:

```
P00>>> show error mem3
Test Directed Errors

No Entries Found

Symptom Directed Errors
  ❶        ❷               ❸            ❹          ❺        ❻         ❼
Entry   Fail Address   Bits/Syndrome   Bank #   ASIC #   Source   Event Type
  0     0be21e00          0cd2           1        1        1          00
  1     0be26b80          0cd2           1        1        1          00
  2     04224020          14,09          2        1        1          01
P00>>>
```

❶ Event log entry number

❷ Fail address—The zero-based module failing address. If the module is configured at base address zero, then the failing address is the offset to the failing DRAM.

❸ Bits/syndrome—First two failing bits (in hexadecimal) for uncorrectible errors; syndrome (in hexadecimal) for correctable errors.

❹ Bank number—The bank number of the failing DRAM.

❺ ASIC number—The ASIC chip that detected the error.

❻ Source—The software or firmware that logged the error.

0—SROM
1—SRM firmware (RBDs)
2—UNIX
3—VMS
4—NT
5–7—Reserved

❼ Event type:

00—Data correctable
01—Data uncorrectable
10—Data uncorrectable (first two bits logged)
11—Other (address and syndrome fields not valid)

CPU Errors

Note

Different CPU types cannot be used within the same system. Example: A KN450 CPU module and a KN460 CPU module cannot be used in the same system.

If an event is logged for any other test than test number 00, the CPU should be replaced. Event logs with just test number 00 do not indicate a bad CPU. Test number 00 indicates that a CPU failover occurred sometime in the past.

All systems must have a CPU module installed in system bus slot 2 (CPU0).

CPU Error Example:

```
P00>>> show error cpu0

CPU0 Module EEROM Event Log

Test Directed Errors
                        ❶
Entry: 0  Test Number: 02       Subtest Number: 02
Parameter 1: 00000000,00000010
Parameter 2: ffffffff,ffffffff
Parameter 3: fffffeff,ffffffff

CPU Event Counters
C3_CA_NOACK     0
   .
   .
   .
C3_DT_PAR_E     0
C3_DT_PAR_O     0

B-Cache Correctable Errors

Entry   Syndrome        Offset L        Offset H        Count

No Entries Found
P00>>>
```

❶ Test Number—A test number other than 00 indicates the CPU should be replaced. Test number 00 indicates a CPU failover has occurred.

3.3.5 clear_error

The `clear_error` command clears error information logged to the serial control bus EEPROMs. The `show fru` command can be used to verify that errors have been cleared (the events logged columns will be set to zeroes).

Synopsis:

clear_error ([all, cpu0–3, mem0–3, io])

Arguments:

[target]	all, CPU{0,1,2,3}, mem{0,1,2,3}, and io.

Examples:

```
P00>>> clear_error all
P00>>>
```

3.3.6 exer_read

The `exer_read` command tests a disk by performing random reads of 2048 bytes on one or more devices. The exercisers are run in the background and nothing is displayed unless an error occurs.

The tests continue until one of the following conditions occurs:

1. All blocks on the device have been read for a passcount of d_passes (default is 1).
2. The exer_read process has been terminated though the `kill` or `kill_diags` commands, or Ctrl/C.
3. The specified time has elapsed.

To terminate the read tests, press Ctrl/C, or use the `kill` command to terminate an individual diagnostic or the `kill_diags` command to terminate all diagnostics. Use the `show_status` display to determine the process ID when terminating an individual diagnostic test.

Synopsis:

exer_read [-sec seconds] [device_name device_name . . .]

Arguments:

[device_name] One or more device names to be tested. The default is du*.*, dk*.*, and dr*.* to test all DSSI and SCSI disks and floppy drives that are on line. These drives may be on the native SCSI bus or connected to an PCI-based controller.

Options:

[-sec seconds] Number of seconds to run exercisers. If you do not enter the number of seconds, the tests will run until d_passes have completed (d_passes default is 1).

If you want to test the entire disk, run at least one pass across the disk. If you do not need to test the entire disk, run the test for 5 or 10 minutes.

Examples:

```
P00>>> exer_read

failed to send command to pkc0.1.0.2.0
failed to send Read to dkc100.1.0.2.0


*** Hard Error - Error #5 -

Diagnostic Name        ID             Device  Pass  Test  Hard/Soft
31-JUL-1992
exer_kid         00000175       dkc100.1.0.2     0     0     1    0
14:54:18
Error in read of 0 bytes at location 014DD400 from device dkc100.1.0.2.0

*** End of Error ***
P00>>>
```

3.3.7 memexer

The memexer command tests memory by running a specified number of memory exercisers. The exercisers are run in the background and nothing is displayed unless an error occurs. Each exerciser tests all available memory in 2 x the backup cache size blocks for each pass.

Note

Certain memory errors that are reported by the OCP may not be reported by the ROM-based diagnostics. Always check the power-up/diagnostic display before running diagnostic commands.

To terminate the memory tests, use the kill command to terminate an individual diagnostic or the kill_diags command to terminate all diagnostics. Use the show_status display to determine the process ID when terminating an individual diagnostic test.

Synopsis:

memexer [number]

Arguments:

[number]	Number of memory exercisers to start. The default is 1. The number of exercisers, as well as the length of time for testing, depends on the context of the testing. Generally, running three to five exercisers for 15 minutes to 1 hour is sufficient for troubleshooting most memory problems.

Examples:

Example with no errors.

```
P00>>> memexer 4
P00>>> show_status

      ID      Program       Device   Pass Hard/Soft Bytes Written    Bytes Read
-------- ------------ ------------ ------ --------- ------------- -------------
00000001         idle       system      0    0    0             0             0
000000c7      memtest       memory      3    0    0     635651584      62565154
000000cc      memtest       memory      2    0    0     635651584      62565154
000000d0      memtest       memory      2    0    0     635651584      62565154
000000d1      memtest       memory      3    0    0     635651584      62565154
P00>>> kill_diags
P00>>>
```

Example with a memory compare error indicating bad memory.

```
P00>>> memexer 4


*** Hard Error - Error #44 - Memory compare error


Diagnostic Name        ID             Device  Pass  Test  Hard/Soft   1-JAN-1995
memtest         000000c8               brd0     1     1     1    0     12:00:01
Expected value:  00000004
Received value:  80000001
Failing addr:    800001c

Failing SIMM module  J32
Failing SIMM module  J31

*** End of Error ***

P00>>> kill_diags
P00>>>
```

3.3.8 memexer_mp

The memexer_mp command tests memory cache coherency in a multiprocessor system by running a specified number of memory exerciser sets. A set is a memory test that runs on each processor checking alternate longwords. The exercisers are run in the background and nothing is displayed unless an error occurs.

Note

Certain memory errors that are reported by the OCP may not be reported by the ROM-based diagnostics. Always check the power-up/diagnostic display before running diagnostic commands.

To terminate the memory tests, use the kill command to terminate an individual diagnostic or the kill_diags command to terminate all diagnostics. Use the show_status display to determine the process ID when terminating an individual diagnostic test.

Synopsis:

memexer_mp [number]

Arguments:

[number] Number of memory exerciser sets to start. The default is 1.

The number of exercisers, as well as the length of time for testing, depends on the context of the testing. Generally, running two or three exercisers for 5 minutes is sufficient.

Examples:

```
P00>>> memexer_mp 2
P00>>> show_status
 ID       Program      Device       Pass  Hard/Soft Bytes Written  Bytes Read
-------- ------------ ------------ ------ --------- ------------- -------------
00000001         idle system            0    0    0             0             0
00000197      memtest memory           50    0    0      51380224      51380224
000001a1      memtest memory           49    0    0      50331648      50331648
000001c2      memtest memory           23    0    0      23068672      23068672
000001cc      memtest memory           19    0    0      18874368      18874368

P00>>> kill_diags
P00>>>
```

3.3.9 nettest

The `nettest` command can be used to run loopback tests for any PCI-based Ethernet ports. It can also be used to test a port on a "live" network.

If the loopback tests are set to run continuously (-p pass_count set to 0), use the `kill` command (or Ctrl/C) to terminate an individual diagnostic or the `kill_diags` command to terminate all diagnostics. Use the `show_status` display to determine the process ID when terminating an individual diagnostic test.

Synopsis:

nettest [-mode port_mode] [-p pass_count] [port]

Arguments:

[port] Specifies the Ethernet port on which to run the test; for example, ewa0 for the DECchip 21040 (TULIP) controller; or era0 for the DEC 4220 (LANCE) controller.

Options:

[-p pass_count] Specifies the number of times to run the test. If 0, then run continuously. The default value is 1. This is the number of passes for the diagnostic. Each pass sends the number of loop messages as set by the environment variable, `era*_loop_count`.

[-mode port_mode] Specifies the mode to set the port adapter.

- ex — external loopback, the default setting (requires a loopback connector or connection to a live network)
- in — internal loopback (loopbacks are conducted within the chip only) Note: Not all network controllers support internal loopback protocol.

Testing an Ethernet Port:

```
P00>>> nettest ewa0 -p 0 &
P00>>> show_status

 ID       Program      Device       Pass  Hard/Soft Bytes Written  Bytes Read
-------- ------------ ------------ ------ --------- ------------- ------------
00000001         idle system            0    0    0             0            0
000000d5      nettest ewa0.0.0.0.0     13    0    0        308672       308672
P00>>> kill_diags
P00>>>
```

Testing an Ethernet Port on a Live Network:

1. Create a list of nodes for which to send MOP loopback packets from port era0.

```
P00>>>echo : 08-00-2B-E2-56-2A > ndbr/lp_nodes_era0
```

2. View the list of nodes.

```
P00>>>P00>>>cat ndbr/lp_nodes_era0
Node: 08-00-2b-e2-56-2a
```

3. Start the testing using the -mode nc flag to leave the port in the default state.

```
P00>>>nettest era0 -mode nc -p 0 &
```

4. View the status of the test.

```
P00>>>show_status

 ID       Program      Device       Pass  Hard/Soft Bytes Written  Bytes Read
-------- ------------ ------------ ------ --------- ------------- -------------
00000001         idle system            0    0    0             0             0
000000b5      nettest era0.0.0.4.1      7    0    0        322068        322000
```

5. Stop the testing.

```
P00>>>kill_diags
P00>>>
```

3.3.10 net -s

The `net -s` command displays the MOP counters for the specified Ethernet port.

Synopsis:

net -s ewa0

Examples:

```
P00>>> net -s ewa0

Status counts:
ti: 72 tps: 0 tu: 47 tjt: 0 unf: 0 ri: 70 ru: 0
rps: 0 rwt: 0 at: 0 fd: 0 lnf: 0 se: 0 tbf: 0
tto: 1 lkf: 1 ato: 1 nc: 71 oc: 0

MOP BLOCK:
 Network list size: 0

MOP COUNTERS:
Time since zeroed (Secs): 42

TX:
 Bytes: 0 Frames: 0
 Deferred: 1 One collision: 0 Multi collisions: 0
TX Failures:
 Excessive collisions: 0 Carrier check: 0 Short circuit: 71
 Open circuit: 0 Long frame: 0 Remote defer: 0
 Collision detect: 71

RX:
 Bytes: 49972 Frames: 70
 Multicast bytes: 0 Multicast frames: 0
RX Failures:
 Block check: 0 Framing error: 0 Long frame: 0
 Unknown destination: 0 Data overrun: 0 No system buffer: 0
 No user buffers: 0
P00>>>
```

3.3.11 net -ic

The `net -ic` command initializes the MOP counters for the specified Ethernet port.

Synopsis:

net -ic ewa0

Examples:

```
P00>>> net -ic ewa0
P00>>> net -s ewa0
Status counts:
ti: 72 tps: 0 tu: 47 tjt: 0 unf: 0 ri: 70 ru: 0
rps: 0 rwt: 0 at: 0 fd: 0 lnf: 0 se: 0 tbf: 0
tto: 1 lkf: 1 ato: 1 nc: 71 oc: 0

MOP BLOCK:
 Network list size: 0

MOP COUNTERS:
Time since zeroed (Secs): 3

TX:
 Bytes: 0 Frames: 0
 Deferred: 0 One collision: 0 Multi collisions: 0
TX Failures:
 Excessive collisions: 0 Carrier check: 0 Short circuit: 0
 Open circuit: 0 Long frame: 0 Remote defer: 0
 Collision detect: 0

RX:
 Bytes: 0 Frames: 0
 Multicast bytes: 0 Multicast frames: 0
RX Failures:
 Block check: 0 Framing error: 0 Long frame: 0
 Unknown destination: 0 Data overrun: 0 No system buffer: 0
 No user buffers: 0
P00>>>
```

3.3.12 kill and kill_diags

The `kill` and `kill_diags` commands terminate diagnostics that are currently executing .

- The `kill` command terminates a specified process.
- The `kill_diags` command terminates all diagnostics.

Synopsis:

kill_diags

kill [PID . . .]

Arguments:

[PID . . .] The process ID of the diagnostic to terminate. Use the `show_status` command to determine the process ID.

3.3.13 show_status

The `show_status` command reports one line of information for each executing diagnostic. The information includes ID, diagnostic program, device under test, error counts, passes completed, and bytes written and read.

Many of the diagnostics run in the background and provide information only if an error occurs. Use the `show_status` command to display the progress of diagnostics.

The following command string is useful for periodically displaying diagnostic status information for diagnostics running in the background:

```
P00>>> while true;show_status;sleep n;done
```

Where *n* is the number of seconds between `show_status` displays.

Synopsis:

show_status

Examples:

```
P00>>> show_status

     ❶          ❷           ❸          ❹     ❺           ❻               ❼
 ID       Program      Device       Pass  Hard/Soft Bytes Written  Bytes Read
-------- ------------ ------------ ------ --------- ------------- -------------
00000001         idle system            0    0    0             0             0
0000006f      memtest memory            1    0    0      35651584      35651584
00000070      memtest memory            1    0    0      35651584      35651584
00000077      memtest memory            1    0    0      37748736      37748736
0000007e     exer_kid dka0.0.0.1.0      0    0    0             0         69120
0000007f     exer_kid dka600.6.0.1      0    0    0             0         66560
00000093     exer_kid dva0.0.0.0.1      0    0    0             0             0
000000d5      nettest ewa0.0.0.0.0     13    0    0        308672        308672
P00>>>
```

❶ Process ID

❷ Program module name

❸ Device under test

❹ Diagnostic pass count

❺ Error count (hard and soft): Soft errors are not usually fatal; hard errors halt the system or prevent completion of the diagnostics.

❻ Bytes successfully written by diagnostic

❼ Bytes successfully read by diagnostic

3.4 Acceptance Testing and Initialization

Perform the following acceptance testing procedure after installing a system or whenever adding or replacing the following:

CPU modules
Memory modules
Standard I/O module
Motherboard
PCI-to-VME daughter board
Storage devices
PCI options

1. Run the RBD acceptance tests using the `test` command.
2. If you have replaced the standard I/O module, run the EISA Configuration Utility (ECU) to remove the default VGA option.
3. Bring up the operating system.
4. Run DEC VET to test that the operating system is correctly installed. Refer to Section 3.5 for information on DEC VET.

3.5 DEC VET

Digital's DEC Verifier and Exerciser Tool (DEC VET) software is a multipurpose system maintenance tool that performs exerciser-oriented maintenance testing. DEC VET runs on OpenVMS AXP and Digital UNIX operating systems. DEC VET consists of a manager and exercisers. The DEC VET manager controls the exercisers. The exercisers test system hardware and the operating system.

DEC VET supports various exerciser configurations, ranging from a single device exerciser to full system loading—that is, simultaneous exercising of multiple devices.

Refer to the *DEC Verifier and Exerciser Tool User's Guide* (AA–PTTMA–TE) for instructions on running DEC VET.

4
Error Log Analysis

This chapter provides information on how to interpret error logs reported by the operating system.

- Section 4.1 describes machine check/interrupts and how these errors are detected and reported.
- Section 4.2 describes the entry format used by the error formatters.
- Section 4.3 describes how to generate a formatted error log using the error formatters available with Digital UNIX and OpenVMS.
- Section 4.4 describes how to interpret the system error log using the bit-to-text translation to isolate the failing FRU.

4.1 Fault Detection and Reporting

Table 4–1 provides a summary of the fault detection and correction components of Digital Alpha 2100 VME systems.

Generally, PALcode handles exceptions as follows:

- The PALcode determines the cause of the exception.
- If possible, it corrects the problem and passes control to the operating system for reporting before returning the system to normal operation.
- If error/event logging is required, control is passed through the system control block (SCB) to the appropriate exception handler.

Table 4–1 Digital Alpha Fault Detection and Correction

Component	Fault Detection/Correction Capability
KN450/KN460 Processor Module	
DECchip 21064 and 21064A microprocessors	Contains error detection and correction (EDC) logic for data cycles. There are check bits associated for all data entering and exiting the 21064 microprocessor. A single-bit error on any of the four longwords being read can be corrected (for each cycle).
Backup cache (B-cache)	EDC check bits on the data store and parity on the tag store and control store.
MS450 Memory Modules	
Memory module	EDC logic protects data by detecting and correcting up to 2 bits for each DRAM chip per gate array. The four bits of data for each DRAM are spread across two gate arrays (one for even longwords, the other for odd longwords).
Standard I/O Module	
I/O module	SCSI controller: SCSI data parity is generated. PCI Ethernet chip: PCI data parity is generated. EISA to PCI bridge chip: PCI data parity is generated. PCI to VME bridge chipset: PCI data parity is generated.
System Bus	
System bus	Longword parity on command, address, and data.
T2 System Bus to PCI Bus Bridge (on Motherboard MBD)	
System bus to PCI bus bridge chip	Longword parity on address and data.

4.1.1 Machine Check/Interrupts

The exceptions that result from hardware system errors are called machine check/interrupts. They occur when a system error is detected during the processing of a data request. There are three types of machine check/interrupts related to system events:

1. Processor machine check
2. System machine check
3. Processor-corrected machine check

The causes of each of the machine check/interrupts are as follows. The system control block (SCB) vector through which PALcode transfers control to the operating system is shown in parentheses.

Processor Machine Check (SCB: 670)

Processor machine check errors are fatal system errors that result in a system crash.

- The DECchip 21064 microprocessor detected one or more of the following uncorrectable data errors:
  - Uncorrectable B-cache data error
  - Uncorrectable memory data error (CU_ERR asserted)
  - Uncorrectable data from other CPU's B-cache (CU_ERR asserted)
- A B-cache tag or tag control parity error occurred
- Hard error was asserted in response to:
  - A system bus read data parity error
  - System bus timeouts (NOACK error bit asserted)—The bus responder detected a write data parity or command address parity error and did not acknowledge the bus cycle.

System Machine Check (SCB: 660)

A system machine check is a system-detected error, external to the DECchip 21064 microprocessor and possibly not related to the activities of the CPU. It occurs when C_ERROR is asserted on the system bus.

Fatal errors:

- The standard I/O module detected a system bus data parity error while serving as system bus commander:
  - System bus errors (NOACK error bit asserted)—The bus responder detected a write data parity or command address parity error and did not acknowledge the bus cycle.
  - Uncorrectable data (CU_ERR asserted) from a responder on the system bus.
  - PCI-reported address data or timeout errors.
- Any system bus device detected a command/address parity error.
- A bus responder detected a write data parity error.
- Memory or standard I/O system bus gate array detected an internal error (SYNC error).

Nonfatal errors:

- A memory module corrected data.
- Correctable B-cache errors were detected while the B-cache was providing data to the system bus (errors from other CPU).
- Duplicate tag store parity errors occurred.

Processor-Corrected Machine Check (SCB: 630)

Processor-corrected machine checks are caused by B-cache errors that are detected and corrected by the DECchip 21064 microprocessor. These are nonfatal errors that result in an error log entry.

4.1.2 System Bus Transaction Cycle

In order to interpret error logs for system bus errors, you need a basic understanding of the system bus transaction cycle and the function of the commander, responder, and bystanders.

For any particular bus transaction cycle there is one commander (either CPU or standard I/O module) that initiates bus transactions and one responder (memory, CPU, or I/O) that accepts or supplies data in response to a command/address from the system bus commander. A bystander is a system bus node (CPU, standard I/O, or memory) that is not addressed by a current system bus commander.

There are four system bus transaction types: Read, write, exchange, and nut.

- Read and write transactions consist of a command/address cycle followed by two data cycles.

- Exchange transactions are used to replace the cache block when a cache block resource conflict occurs. They consist of a command/address cycle followed by four data cycles: Two writes and two reads.
- Nut transactions consist of a command/address cycle and two dummy data cycles for which no data is transferred.

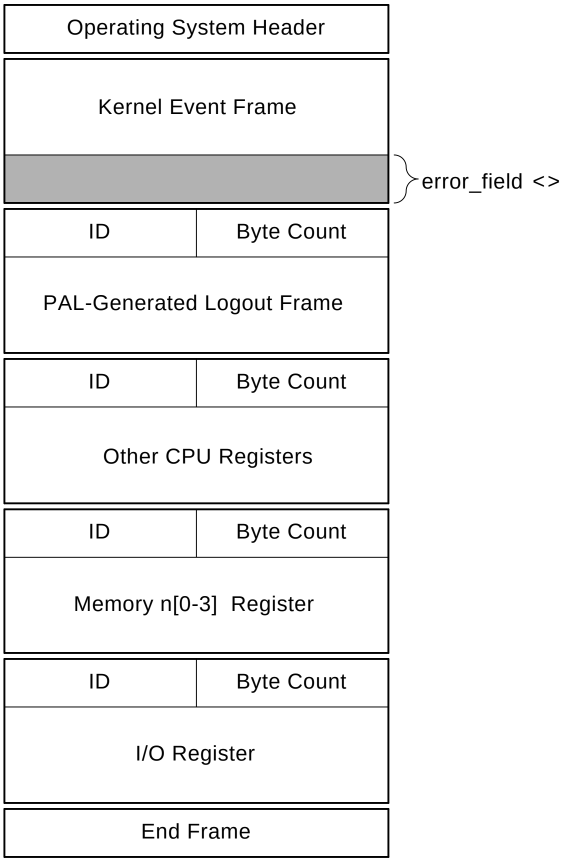
4.2 Error Logging and Event Log Entry Format

The Digital UNIX and OpenVMS error handlers can generate several entry types. All error entries, with the exception of correctable memory errors, are logged immediately. Entries can be of variable length based on the number of registers within the entry.

Each entry consists of an operating system header, kernel event frame, several device frames, and an end frame. Most entries have a PAL-generated logout frame, and may contain other CPU frames (0–3), memory (0–3), and I/O.

Figure 4–1 shows the general error log format used by the DECevent, ERF, and uerf error formatters.

Figure 4–1 Error Log Format



▨ The 128-bit error field is the primary field for isolating system kernel faults.

LJ-02628-TI0

By examining the error fields (0–3) of the kernel event frame, you can isolate the failing system FRU for system faults reported by the operating system. One or more bits are set in the error fields as the result of the system error handling process. During the error handling process, errors are first handled by the appropriate PALcode error routine and then by the associated operating system error handler.

Section 4.4 describes how to interpret the error field to isolate to the FRU that is the source of the failure. Forthcoming fault management and error notification tools will key off of these error field bits.

4.3 Event Record Translation

Error formatters translate the entry into the format described in Section 4.2.

- Systems running OpenVMS can use the DECevent and ERF error formatters.
- Systems running Digital UNIX uses the uerf error formatter.

DECevent, ERF, and uerf provide bit-to-text translation for the kernel event frame.

- Section 4.3.1 summarizes the commands used to translate the error log information for the OpenVMS operating system using DECevent.
- Section 4.3.2 summarizes the commands used to translate the error log information for the OpenVMS operating system using ERF.
- Section 4.3.3 summarizes the commands used to translate the error log for the Digital UNIX operating system.

4.3.1 OpenVMS Translation Using DECevent

The kernel error log entries are translated from binary to ASCII using the DIAGNOSE command. To invoke the error log utility, enter the DCL command DIAGNOSE.

Format:

DIAGNOSE/TRANSLATE [*qualifier*] [, . . .] [*infile*[, . . .]]

Example:

```
$ DIAGNOSE/TRANSLATE/SINCE=14-JUN-1994
```

For more information on generating error log reports using DECevent, refer to the *DECevent Translation and Reporting Utility for OpenVMS*.

DECevent bit-to-text translation highlights all error fields that are set and other significant states. These are displayed in capital letters in the third column of the error log (see ❸ in Example 4–1). Otherwise, nothing is shown in the translation column.

Section 4.4.6 provides a sample DECevent-generated error log.

4.3.2 OpenVMS Translation Using ERF

The kernel error log entries are translated from binary to ASCII using the ANALYZE/ERROR command. To invoke the error log utility, enter the DCL command ANALYZE/ERROR_LOG.

Format:

ANALYZE_ERROR_LOG [/*qualifier(s)*] [*file-spec*] [, . . .]

Example:

```
$ ANALYZE/ERROR_LOG/INCLUDE=(CPU,MEMORY)/SINCE=TODAY
```

As shown in the above example, the OpenVMS error handler also supports the /INCLUDE qualifier, such that CPU and memory error entries can be translated selectively.

ERF bit-to-text translation highlights all error fields that are set, and other significant state. These are displayed in capital letters in the third column of the error log. Otherwise, nothing is shown in the translation column.

4.3.3 Digital UNIX Translation Using uerf

Error log information is written to /var/adm/binary.errlog. Use the following command to save the error log information by copying it to another file:

```
$ cp /var/adm/binary.errlog /tmp/errors_upto_today
```

To clear the error log file, use the following command:

```
$ cp /dev/null /var/adm/binary.errlog
```

To produce a bit-to-text translation of the error log file, use the following command:

```
$ uerf -f /tmp/errors_upto_today -R
```

To view all all error logs in reverse chronological order, use the following command:

```
$ uerf -R
```

For filtering of error logs, see the reference page (man page) for uerf on the system you are currently using:

```
$ man uerf
```

4.4 Interpreting System Faults

Use the following steps to determine the failing FRU when a system error is reported through an error log.

1. Examine the error fields of the kernel event frame.

   If a system error has been reported, one or more bits may be set for the error fields, 0–3, and their corresponding bit-to-text definition will be listed.

2. Using Table 4–2, find the entry that matches the set bit and bit-to-text to determine the most probable source of the fault listed in the third column. The field replaceable units (FRUs) for the core system are listed as follows:

   CPUnn — CPU module (0–3)
   MEMnn — Memory module (0–3)
   I/O_0 — Standard I/O module
   PCInn — PCI modules (0–2)
   MBD — System bus motherboard, which contains the T2, system bus to PCI bus bridge chip.

3. If the table entry lists a note number along with the most probable failing module, refer to that note following Table 4–2.

   There are five possible notes, Note 1 through Note 5. Each note provides a synopsis of the problem and additional information to consider for analysis.

Section 4.4.6 provides a sample DECevent-generated error log.

Table 4–2 Error Field Bit Definitions for Error Log Interpretation

Error Field Bits	Bit-to-Text Definition	Module/Notes
Quadword 0, error_field_0<63:0>, CPU0-Detected		
W0-Byte-0, CPU Machine Check Related Errors		
<0> C3_0_CA_NOACK	CPU_0 Bus Command No-Ack	CPU_0, Note 1
<1> C3_0_WD_NOACK	CPU_0 Bus Write Data No-Ack	CPU_0, Note 2
<2> C3_0_RD_PAR	CPU_0 Bus Read Data Parity Error	CPU_0, Note 3
<3> EV_0_C_UNCORR	CPU_0 Cache Uncorrectable	CPU_0, Note 4
<4> EV_0_TC_PAR	CPU_0 Cache Tag Control Parity Error	CPU_0
<5> EV_0_T_PAR	CPU_0 Cache Tag Parity Error	CPU_0
<6> C3_0_EV	CPU_0 CPU to System Bus Interface Data Error	CPU_0
<7> C3_0_RETRY_ FAILED	CPU_0 Retry Failed	CPU_0
W0-Byte-1, CPU Interrupt and Machine Check Related Errors		
<0> C3_0_C_UNCORR	CPU_0 Cache Uncorrectable (system bus interface detected)	CPU_0, Note 4
<1> C3_0_TC_PAR	CPU_0 Cache Tag Control Parity Error	CPU_0
<2> C3_0_T_PAR	CPU_0 Cache Tag Parity Error	CPU_0
<3> C3_0_C_CORR	CPU_0 Cache Correctable (system bus interface detected)	CPU_0
<4> EV_0_C_CORR	CPU_0 Cache Correctable (CPU detected)	CPU_0
<5> C3_0_SYN_1F	CPU_0 Cache Uncorrectable (0x1f Syndrome)	CPU_0

(continued on next page)

Table 4–2 (Cont.) Error Field Bit Definitions for Error Log Interpretation

Error Field Bits	Bit-to-Text Definition	Module/Notes
Quadword 1, CPU1-Detected		
W1-Byte-0, CPU Machine Check Related Errors		
<0> C3_1_CA_NOACK	CPU_1 Bus Command No-Ack	CPU_1, Note 1
<1> C3_1_WD_NOACK	CPU_1 Bus Write Data No-Ack	CPU_1, Note 2
<2> C3_1_RD_PAR	CPU_1 Bus Read Parity Error	CPU_1, Note 3
<3> EV_1_C_UNCORR	CPU_1 Cache Uncorrectable (CPU detected)	CPU_1, Note 4
<4> EV_1_TC_PAR	CPU_1 Cache Tag Control Parity Error	CPU_1
<5> EV_1_T_PAR	CPU_1 Cache Tag Parity Error	CPU_1
<6> C3_1_EV	CPU_1 CPU to System Bus Interface Data Error	CPU_1
<7> C3_1_RETRY_ FAILED	CPU_1 Retry failed	CPU_1, MBD, or PCI target
W1-Byte-1, CPU Interrupt and Machine Check Related Errors		
<0> C3_1_C_UNCORR	CPU_1 Cache Uncorrectable (system bus interface detected)	CPU_1, Note 4
<1> C3_1_TC_PAR	CPU_1 Cache Tag Control Parity Error	CPU_1
<2> C3_1_T_PAR	CPU_1 Cache Tag Parity Error	CPU_1
<3> C3_1_C_CORR	CPU_1 Cache Correctable (system bus interface detected)	CPU_1
<4> EV_1_C_CORR	CPU_1 Cache Correctable (CPU detected)	CPU_1
<5> C3_1_SYN_1F	CPU_1 Cache Uncorrectable (0x1f Syndrome)	CPU_1

(continued on next page)

Table 4–2 (Cont.) Error Field Bit Definitions for Error Log Interpretation

Error Field Bits	Bit-to-Text Definition	Module/Notes
Quadword 2, CPU2-Detected		
W2-Byte-0, CPU Machine Check Related Errors		
<0> C3_2_CA_NOACK	CPU_2 Bus Command No-Ack	CPU_2, Note 1
<1> C3_2_WD_NOACK	CPU_2 Bus Write Data No-Ack	CPU_2, Note 2
<2> C3_2_RD_PAR	CPU_2 Bus Read Parity Error	CPU_2, Note 3
<3> EV_2_C_UNCORR	CPU_2 Cache Uncorrectable (CPU detected)	CPU_2, Note 4
<4> EV_2_TC_PAR	CPU_2 Cache Tag Control Parity Error	CPU_2
<5> EV_2_T_PAR	CPU_2 Cache Tag Parity Error	CPU_2
<6> C3_2_EV	CPU_2 CPU to System Bus Interface Data Error	CPU_2
<7> C3_2_RETRY_ FAILED	CPU_2 Retry failed	CPU_2
W2-Byte-1, CPU Interrupt and Machine Check Related Errors		
<0> C3_2_C_UNCORR	CPU_2 Cache Uncorrectable (system bus interface detected)	CPU_2, Note 4
<1> C3_2_TC_PAR	CPU_2 Cache Tag Control Parity Error	CPU_2
<2> C3_2_T_PAR	CPU_2 Cache Tag Parity Error	CPU_2
<3> C3_2_C_CORR	CPU_2 Cache Correctable (system bus interface detected)	CPU_2
<4> EV_2_C_CORR	CPU_2 Cache Correctable (CPU detected)	CPU_2
<5> C3_2_SYN_1F	CPU_2 Cache Uncorrectable (0x1f Syndrome)	CPU_2

(continued on next page)

Table 4–2 (Cont.) Error Field Bit Definitions for Error Log Interpretation

Error Field Bits	Bit-to-Text Definition	Module/Notes
Quadword 3, CPU3-Detected		
W3-Byte-0, CPU Machine Check Related Errors		
<0> C3_3_CA_NOACK	CPU_3 Bus Command No-Ack	CPU_3, Note 1
<1> C3_3_WD_NOACK	CPU_3 Bus Write Data No-Ack	CPU_3, Note 2
<2> C3_3_RD_PAR	CPU_3 Bus Read Parity Error	CPU_3, Note 3
<3> EV_3_C_UNCORR	CPU_3 Cache Uncorrectable (CPU detected)	CPU_3, Note 4
<4> EV_3_TC_PAR	CPU_3 Cache Tag Control Parity Error	CPU_3
<5> EV_3_T_PAR	CPU_3 Cache Tag Parity Error	CPU_3
<6> C3_3_EV	CPU_3 CPU to System Bus Interface Data Error	CPU_3
<7> C3_3_RETRY_ FAILED	CPU_3 Retry failed	CPU_3
W3-Byte-1, CPU Interrupt and Machine Check Related Errors		
<0> C3_3_C_UNCORR	CPU_3 Cache Uncorrectable (system bus interface detected)	CPU_3, Note 4
<1> C3_3_TC_PAR	CPU_3 Cache Tag Control Parity Error	CPU_3
<2> C3_3_T_PAR	CPU_3 Cache Tag Parity Error	CPU_3
<3> C3_3_C_CORR	CPU_3 Cache Correctable (system bus interface detected)	CPU_3
<4> EV_3_C_CORR	CPU_3 Cache Correctable (CPU detected)	CPU_3
<5> C3_3_SYN_1F	CPU_3 Cache Uncorrectable (0x1f Syndrome)	CPU_3

(continued on next page)

Table 4–2 (Cont.) Error Field Bit Definitions for Error Log Interpretation

Error Field Bits	Bit-to-Text Definition	Module/Notes
Quadword 1, error_field_1<63:0> — I/O as Commander (bus errors that the T2 system bus to EISA bus bridge chip can detect while the I/O module is Commander)		
W0-Byte-0, External Cause		
<0> IO_CA_NOACK	T2 detected Bus Command/Add No-Ack	MBD or responder, Note 1
<1> IO_WD_NOACK	T2 detected Bus Write Data No-Ack	MBD or responder, Note 2
<2> IO_RD_PAR	T2 detected Bus Read Parity Error	MBD or target, Note 3
<3> IO_CB_UNCORR	Data received by T2 is corrupted	Target, Note 5
W0-Byte-1, Internal Cause		
<0> PCI_WR_PAR	T2 - PCI Write Data Parity Error	I/O
<1> PCI_ADD_PAR	T2 - PCI Address Parity Error	I/O
<2> PCI_RD_PAR	T2 - PCI Read Data Parity Error	I/O
<3> PCI_DEV_PAR	T2 - PCI Parity Error	I/O
<4> PCI_SYS_ERR	T2 - PCI System Error	I/O
<5> PCI_TIMEOUT	T2 - PCI Timeout Error	I/O
<6> PCI_NMI	T2 - PCI NMI Asserted	I/O

(continued on next page)

Table 4–2 (Cont.) Error Field Bit Definitions for Error Log Interpretation

Error Field Bits	Bit-to-Text Definition	Module/Notes
Quadword 1, error_field_0<63:0>, Responder Errors		
W0-Byte-0, Command/Address Parity Error Detected		
<0> C3_0_CA_PAR	CPU_0 Command/Address Parity Error	CPU_0, Note 1
<1> C3_1_CA_PAR	CPU_1 Command/Address Parity Error	CPU_1, Note 1
<2> MEM0_CA_PAR	MEM_0 Command/Address Parity Error	MEM_0, Note 1
<3> MEM1_C3_2_CA_PAR	MEM_1 or CPU_2 Command/Address Parity Error	MEM_1, CPU2, Note 1
<4> MEM2_CA_PAR	MEM_2 Command/Address Parity Error	MEM_2, Note 1
<5> MEM3_CA_PAR	MEM_3 Command/Address Parity Error	MEM_3, Note 1
<6> IO_CA_PAR	I/O Command/Address Parity Error	I/O_0, Note 1
<7> EXT_IO_C3_3_CA_PAR	External I/O or CPU3 Command/Address Parity Error	I/O_1, CPU3, Note 1
W0-Byte-1, System Bus Interface Write Data Parity Errors		
<0> C3_0_WD_PAR	CPU_0 Write Data Parity Error	CPU_0, Note 2
<1> C3_1_WD_PAR	CPU_1 Write Data Parity Error	CPU_1, Note 2
<2> MEM0_WD_PAR	MEM_0 Write Data Parity Error	MEM_0, Note 2
<3> MEM1_C3_2_WD_PAR	MEM_1 or CPU2 Write Data Parity Error	MEM_1, CPU2 Note 2
<4> MEM2_WD_PAR	MEM_2 Write Data Parity Error	MEM_2, Note 2
<5> MEM3_WD_PAR	MEM_3 Write Data Parity Error	MEM_3
<6> IO_WD_PAR	I/O Write Data Parity Error	I/O_0
<7> EXT_IO_C3_3_WD_PAR	External I/O Write Data Parity Error	I/O_1

(continued on next page)

Table 4–2 (Cont.) Error Field Bit Definitions for Error Log Interpretation

Error Field Bits	Bit-to-Text Definition	Module/Notes
W1-Byte-0, Memory Uncorrectable Errors		
<0> MEM0_UNCORR	MEM_0 Uncorrectable Error	MEM_0
<1> MEM1_UNCORR	MEM_1 Uncorrectable Error	MEM_1
<2> MEM2_UNCORR	MEM_2 Uncorrectable Error	MEM_2
<3> MEM3_UNCORR	MEM_3 Uncorrectable Error	MEM_3
W1-Byte-1, Memory Correctable Errors		
<0> MEM0_CORR	MEM_0 Correctable Error	MEM_0
<1> MEM1_CORR	MEM_1 Correctable Error	MEM_1
<2> MEM2_CORR	MEM_2 Correctable Error	MEM_2
<3> MEM3_CORR	MEM_3 Correctable Error	MEM_3
<4> MEM0_COR_DIS	MEM_0 EDC Correction Disabled	MEM_0
<5> MEM1_COR_DIS	MEM_1 EDC_Correction Disabled	MEM_1
<6> MEM2_COR_DIS	MEM_2 EDC_Correction Disabled	MEM_2
<7> MEM3_COR_DIS	MEM_3 EDC_Correction Disabled	MEM_3
W2-Byte-0, Sync Errors (the two gate arrays are not working together)		
<0> MEM0_SYNC_Error	MEM_0 Chip Sync Error	MEM_0
<1> MEM1_SYNC_Error	MEM_1 Chip Sync Error	MEM_1
<2> MEM2_SYNC_Error	MEM_2 Chip Sync Error	MEM_2
<3> MEM3_SYNC_Error	MEM_3 Chip Sync Error	MEM_3
<4> IO_BUSSYNC	I/O Module System Bus Sync Error	MBD

(continued on next page)

Table 4–2 (Cont.) Error Field Bit Definitions for Error Log Interpretation

Error Field Bits	Bit-to-Text Definition	Module/Notes
Miscellaneous Flags		
W3-Byte-0, CPU-Specific (in context of CPU that is reporting the error)		
<0> EV_SYN_1F	CPU Reported Syndrome 0x1f	Note 4
<1> Reserved	Reserved SBZ	
<2> DT_PAR	Duplicate Tag Store Parity Error	This CPU
<3> EV_HARD_ERROR	CPU Cycle Aborted with HARD ERROR	
W3-Byte-1, Event Correlation Flags		
<0> C3_MEM_R_ERROR	CPU error caused by memory	Note 4
<1> IO_MEM_R_ERROR	I/O error caused by memory	
<2> C3_OCPU_ADD_ MATCH	CPU error caused by other CPU	Note 4
<3> MIXED_ERRORS	Mixed errors (no correlation)	
Quadword 3, error_field_3<63:0>, PCI and EISA Errors		
W0-Byte-0, PCI 0 Status Reported Errors		
<0> PCI_0_Parr_ Detected_as_Master	PCI 0 Data Parr Detected While Bus Master	PCI0 or MBD
<1> PCI_0_SIG_Target_ Abort	PCI 0 Aborted with Target-Abort While Target	PCI0 or MBD
<2> PCI_0_REC_Target_ Abort	PCI 0 Received Target-Abort While Target	PCI0 or MBD
<3> PCI_0_REC_Master_ Abort	PCI 0 Cycle Terminated with Master-Abort While Master	PCI0 or MBD
<4> PCI_0_SIG_System_ Error	PCI 0 Signaled System Error	PCI0 or MBD
<5> PCI_0_Detected_ Parity_Error	PCI 0 Detected Parity Error	PCI0 or MBD

(continued on next page)

Table 4–2 (Cont.) Error Field Bit Definitions for Error Log Interpretation

Error Field Bits	Bit-to-Text Definition	Module/Notes
W0-Byte-1, PCI 1 Status Reported Errors		
<0> PCI_1_Parr_ Detected_as_Master	PCI 1 Data Parr Detected While Bus Master	PCI1 or MBD
<1> PCI_1_SIG_Target_ Abort	PCI 1 Aborted with Target-Abort While Target	PCI1 or MBD
<2> PCI_1_REC_Target_ Abort	PCI 1 Received Target-Abort While Master	PCI1 or MBD
<3> PCI_1_REC_Master_ Abort	PCI 1 Cycle Terminated with Master-Abort While Master	PCI1 or MBD
<4> PCI_1_SIG_System_ Error	PCI 1 Signaled System Error	PCI1 or MBD
<5> PCI_1_Detected_ Parity_Error	PCI 1 Detected Parity Error	PCI1 or MBD
W1-Byte-0, PCI 2 Status Reported Errors		
<0> PCI_2_Parr_ Detected_as_Master	PCI 2 Data Parr Detected While Bus Master	PCI2 or MBD
<1> PCI_2_SIG_Target_ Abort	PCI 2 Aborted with Target-Abort While Target	PCI2 or MBD
<2> PCI_2_REC_Target_ Abort	PCI 2 Aborted with Target-Abort While Master	PCI2 or MBD
<3> PCI_2_REC_Master_ Abort	PCI 2 Cycle Terminated with Master-Abort While Master	PCI2 or MBD
<4> PCI_2_SIG_System_ Error	PCI 2 Signaled System Error	PCI2 or MBD
<5> PCI_2_Detected_ Parity_Error	PCI 2 Detected Parity Error	PCI2 or MBD

(continued on next page)

Table 4–2 (Cont.) Error Field Bit Definitions for Error Log Interpretation

Error Field Bits	Bit-to-Text Definition	Module/Notes
W2-Byte-0, PCI Ethernet Chip (TULIP) Status Reported Errors		
<0> TULIP_PARR_ Detected_as_Master	TULIP Parr Detected While Bus Master	I/O_0
<1> TULIP_SIG_Target_ Abort	TULIP Aborted with Target-Abort While Target	I/O_0
<2> TULIP_REC_Target_ Abort	TULIP Aborted with Target-Abort While Master	I/O_0
<3> TULIP_REC_Master_ Abort	TULIP Cycle Terminated with Master-Abort While Master	I/O_0
<4> TULIP_SIG_System_ Error	TULIP Signaled System Error	I/O_0
<5> TULIP_Detected_ Parity_Error	TULIP Detected Parity Error	I/O_0
W2-Byte-1, SCSI Controller (NCR) Status Status Reported Errors		
<0> NCR_PARR_ Detected_as_Master	NCR Data Parr Detected While Bus Master	I/O_0
<1> NCR_SIG_Target_ Abort	NCR Aborted with Target-Abort While Target	I/O_0
<2> NCR_REC_Target_ Abort	NCR Aborted with Target-Abort While Master	I/O_0
<3> NCR_REC_Master_ Abort	NCR Cycle Terminated with Master-Abort While Master	I/O_0
<4> NCR_SIG_System_ Error	NCR Signaled System Error	I/O_0
<5> NCR_Detected_ Parity_Error	NCR Detected Parity Error	I/O_0

(continued on next page)

Table 4–2 (Cont.) Error Field Bit Definitions for Error Log Interpretation

Error Field Bits	Bit-to-Text Definition	Module/Notes
W3-Byte-0, PCI–EISA Bridge, PCI Status Reported Errors		
<0> PCEB_PARR_ Detected_As_Master	PCEB Data Parr Detected While Bus Master	I/O_0
<2> PCEB_SIG_Target_ Abort	PCEB Aborted with Target-Abort While Target	I/O_0
<3> PCEB_REC_Master_ Abort	PCEB Cycle Terminated with Master-Abort While Master	I/O_0
<4> PCEB_SIG_System_ Error	PCEB Signaled System Error	I/O_0
<5> PCEB_Detected_ Parity_Error	PCEB Detected Parity Error	I/O_0
W3-Byte-1, EISA System Component (ESC) Reported Errors		
<0> ESC_PCI_PERR_ Detected	ESC Detected PCI Perr	I/O_0
<1> ESC_EISA_Timeout	ESC Detected EISA Bus Time-Out	I/O_0
<2> ESC_EISA_IOCHK	ESC Detected EISA IOCHK	I/O_0
<3> ESC_FAIL-SAFE	ESC Fail-Safe Timer Expired	I/O_0

4.4.1 Note 1: System Bus Address Cycle Failures

Synopsis:

System bus address cycle failures can be reported by the bus commander, responders, or both:

- By commander: _CA_NOACK—Bus Command Address No-Ack

  Commander did not receive an acknowledgment command/address. Probable causes are:

  – A programming error, software fault (addressed nonexistent address)

  – A bus buffer failure on the bus commander

- By responders: _CA_PAR—Bus Command/Address Parity Error

  Responder detected a parity error during the command/address cycle. The bus was corrupted by commander module (I/O or CPU), backplane, or responder module (I/O, memory, or CPU).

Analysis:

Note

All bus nodes check command/address parity during the command/address cycle.

- _CA_NOACK errors without respective command/address parity errors are most likely caused by problems in the bus commander, such as programming errors, address generation, and the like. You should consider the context of the error: For example, a software fault may cause the system to crash each time you run a particular piece of software.
- _CA_NOACK errors with all responders reporting command/address parity errors are most likely caused by a bus commander failure or bus failure.
- _CA_PAR errors, without respective command/address NOACKs, are most likely the result of a failing buffer within the device reporting the isolated CA_PAR error.

4.4.2 Note 2: System Bus Write-Data Cycle Failures

Synopsis:

System Bus Write Data failures can be reported by the bus commander, responders, or both.

- By commander: _WD_NOACK—Write-Data No-Ack

  Commander did not receive an acknowledgment to write-data cycle. A bus buffer failure on the bus commander is the probable cause.
- By responders: _WD_PAR—Write-Data Parity Error

  Responder detected a parity error during the write-data cycle. The bus was corrupted by commander module (I/O or CPU), backplane, or responder module (I/O, memory, or CPU).

Analysis:

Note

Only the addressed bus responder checks write-data parity.

- _WD_NOACK (write-data NOACK) errors without respective WD_PAR (write-data parity) errors are most likely caused by problems in the bus commander. However, there is a small probability that the responder could be at fault.

Examine the commander’s command trap register to identify the respective responder.

- _WD_NOACK errors with the responder reporting _WD_PAR errors could indicate a failure with either device.
- _WD_PAR errors without respective _WD_NOACK would require two failures to occur:

  1. Bad data received by responder
  2. A valid response was received when one should not have been sent.

  The failing module could be either partner in the transfer.

4.4.3 Note 3: System Bus Read Parity Error

Synopsis:

System bus read-data failures are reported only by the bus commander.

- By commander: _RD_PAR error—Read-data parity error.

  The bus commander (device reporting _RD_PAR) detected a parity error on data received from the system bus.

Analysis:

Note

Only the bus commander checks write-data parity on bus reads.

- The failure could be caused by either the bus commander or responder. The failing data’s address is captured in the commander’s bus trap register.
- A system bus read parity error can result as a side effect of a command/address NOACK.

4.4.4 Note 4: Backup Cache Uncorrectable Error

Synopsis:

Data from the backup cache is either delivered to the DECchip 21064 microprocessor or the system bus interface chip is corrupted.

Analysis:

The failing module is the CPU reporting the failure, except:

- If EV_SYN_1F (“CPU reported syndrome 0x1f”) or C3_SYN_1F (“C3 reported syndrome 0x1f”) bits are set in the error field, known bad data was supplied to the CPU from another source (either memory or the other CPU).
  - If C3_MEM_R_ERROR (“CPU error caused by memory”) bit is set, examine MEMn_UNCORR (“MEM_n Uncorrectable Error”) or MEMn_SYNC_Error (“MEM_n Chip Sync Error”) to identify which memory was the source of the error.
  - If C3_OCPU_ADD_MATCH (“CPU error caused by other CPU”) is set, the other CPU caused the error.
- If other error bits associated with the CPU reporting the error are also set, it is likely that the fault is associated with this CPU module.

4.4.5 Note 5: Data Delivered to I/O Is Known Bad

Synopsis:

IO_CB_UNCORR—I/O module received data identified as bad from system bus.

Analysis:

Check to see if the following bits are set for the error field:

MEMn_UNCORR (“MEM_n Uncorrectable Error”)
MEMn_SYNC_Error (“MEM_n Chip Sync Error”)
CPUn_XXXXXX errors (“CPU_n xxx... error”)

4.4.6 Sample System Error Report (DECevent)

Example 4–1 provides an abbreviated DECevent-generated error log entry for a processor machine check, SCB 670 (❶). Error field 0 (❷), has one bit set. The corresponding bit-to-text translation (❸) is provided in the third column.

Example 4–1 DECevent-Generated Error Log Entry Indicating CPU Error

```
******************************** ENTRY    1 ********************************

Logging OS                        3. OpenVMS AXP
OS version                           X5OJ-FT3
Event sequence number             2.
Timestamp of occurrence              22-JUN-1994 17:46:03
System uptime in seconds         79.
VMS error mask            x00000000
VMS flags                     x0000
Host name
AXP HW model                         AlphaServer 2100
System type register      x00000009
Unique CPU ID             x00000002
mpnum                     x000000FF
mperr                     x000000FF

Event validity                   -1. Unknown validity code
Event severity                   -1. Unknown severity code
Entry type                        2.
Major Event class                 1. CPU
AXP Device Type                   0.

CPU Minor class                   1. Machine check (670 entry) ❶

Entry Byte Count          x00000420
frame revision                x0000
scb vector                    x0670  processor machine check
severity                      x0000  field not valid
cpu id                        x0000
error count                   x0001
fail code                     x0000  field not valid
error field0 ❷           x0000000000000005
                                     CPU_0 Bus Command No-Ack ❸
                                     CPU_0 Bus Parity Error
error field1              x0000000000000000
error field2              x0008000000000000
                                     CPU cycle aborted with HARD ERROR
error field3              x0000000000000000

frame id                  x00000002  MC 670 Frame
        .
        .
        .
```

5 System Configuration and Setup

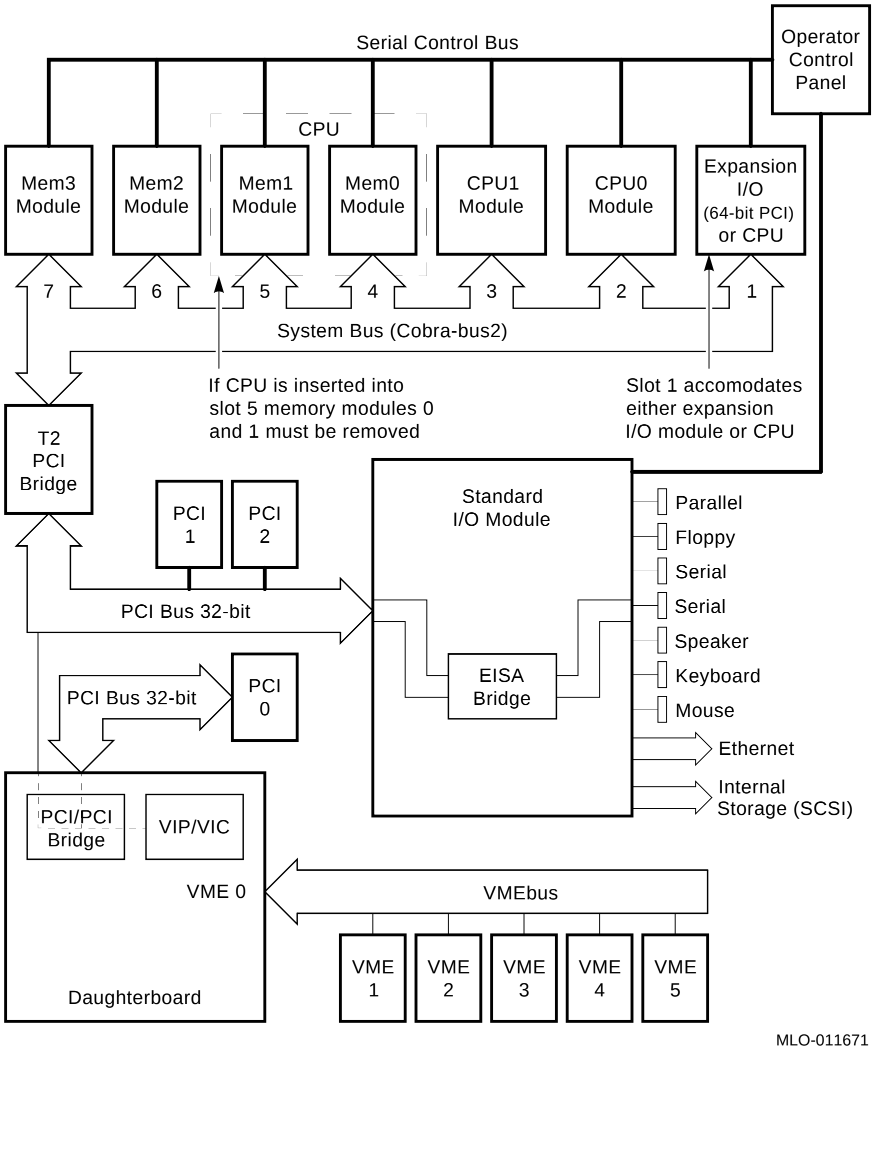
This chapter provides configuration and setup information for Digital Alpha 2100 VME systems and system options.

- Section 5.1 describes how to examine the system configuration using the console firmware.
  - Section 5.1.1 describes the function of the two firmware interfaces used with Alpha systems.
  - Section 5.1.2 describes how to switch between firmware interfaces.
  - Section Section 5.1.3 describes the commands used to examine the system configuration.
- Section 5.2 describes the system bus configuration.
- Section 5.3 describes the function of the standard I/O module.
- Section 5.4 describes the PCI bus.
- Section 5.5 describes the VME bus.
- Section 5.6 describes how to configure and install SCSI drives in the system.
- Section 5.7 describes the console port configurations.

5.1 Verifying System Configuration

Figure Figure 5–1 illustrates the system architecture for Digital Alpha 2100 VME systems.

Figure 5–1 System Architecture for the Digital Alpha VME 2100 (BA742 Enclosure)



5.1.1 System Firmware

At product introduction, system firmware provides support for the following operating systems:

- Digital UNIX
- OpenVMS Alpha

Digital UNIX and OpenVMS Alpha are supported under the SRM command line interface, which can be serial or graphical. The SRM firmware is in compliance with the *Alpha System Reference Manual* (SRM).

The console firmware provides the data structures and callbacks available to booted programs defined in both the SRM standards.

SRM Command Line Interface

Systems running Digital UNIX or OpenVMS access the SRM firmware via a command line interface (CLI). The CLI is a UNIX-style shell that provides a set of commands and operators as well as a scripting facility. It allows you to configure and test the system, examine and alter the system state, and boot the operating system.

The only task that you cannot perform from the SRM command line interface is run the EISA Configuration Utility (ECU) or Raid Configuration Utility (RCU). To run the ECU, you must enter the `ecu` command. This will boot the ARC firmware and the ECU software. For Digital Alpha 2100 VME systems, you need to run the ECU when replacing the standard I/O module. The replacement standard I/O is shipped with EISA configuration data for a VGA controller used in AlphaServer 2000/2100 systems (Sable systems).

5.1.2 Switching Between Interfaces

For a few procedures it is necessary to switch from one console interface to the other.

- All console tests are run from the SRM interface.
- The EISA Configuration Utility (ECU) and the RAID Configuration Utility (RCU) are run from the ARC interface.

Switching from SRM to ARC

Two SRM console commands are used to switch to the ARC console:

- The `arc` command loads the ARC firmware and switches to the ARC menu interface.
- The `ecu` command loads the ARC firmware and then boots the ECU diskette.

Switching from ARC to SRM

Switch from the ARC console to the SRM console as follows:

1. From the Boot menu, select the Supplementary menu.
2. From the Supplementary menu, select “Set up the system.”
3. From the Setup menu, select “Switch to OpenVMS or UNIX console.” This allows you to select your operating system console.
4. Select your operating system, then press enter on “Setup menu.”
5. When the message “Power-cycle the system to implement the change” is displayed, press the Reset button. Once the console firmware is loaded and device drivers are initialized, you can boot the operating system.

5.1.3 Verifying Configuration: SRM Console Commands for Digital UNIX and OpenVMS

The following SRM console commands are used to verify system configuration on Digital UNIX and OpenVMS systems:

- `show config` (Section 5.1.3.1)—Displays the buses on the system and the devices found on those buses.
- `show device` (Section 5.1.3.2)—Displays the devices and controllers in the system.
- `show memory` (Section 5.1.3.3)—Displays main memory configuration.
- `set` and `show` (Section 5.1.3.4)—Set and display environment variable settings.

5.1.3.1 show config

The show config command displays all devices found on the system bus, PCI bus, and EISA bus. You can use the information in the display to identify target devices for commands such as boot and test, as well as to verify that the system sees all the devices that are installed.

The configuration display includes the following:

- Core system status:

  CPU, memory, standard I/O are shown with the results of power-up tests: P (pass) or F (fail)

- Hose 0, Bus 0, PCI:
  - Slot 0 = Ethernet adapter (ewa0)
  - Slot 1 = SCSI controller on standard I/O, along with storage drives on the bus.
  - Slot 2 = EISA to PCI bridge chip
  - Slots 3–5 = Reserved
  - Slot 6 = PCI to PCI bridge chip (on VME daughter board)
  - Slot 7 = Corresponds to PCI slot 7 (PCI7)
  - Slot 8 = Corresponds to PCI slot 8 (PCI8)

  In the case of storage controllers, the devices off the controller are also displayed.

- Hose 0, Bus 1, PCI

  Slot 0 = PCI to VME chip (DECchip 7407)
  Slot 1 = corresponds to PCI slot 1 (PCI1)

  In the case of storage controllers, the devices off the controller are also displayed.

- Hose 1, Bus 0, EISA:

  Not applicable to Digital Alpha 2100 VME systems.

- Hose 2, Bus 0, PCI:

  Reserved for future expansion.

For more information on device names, refer to Section 5.1.3.2, show device.

Synopsis:

show config

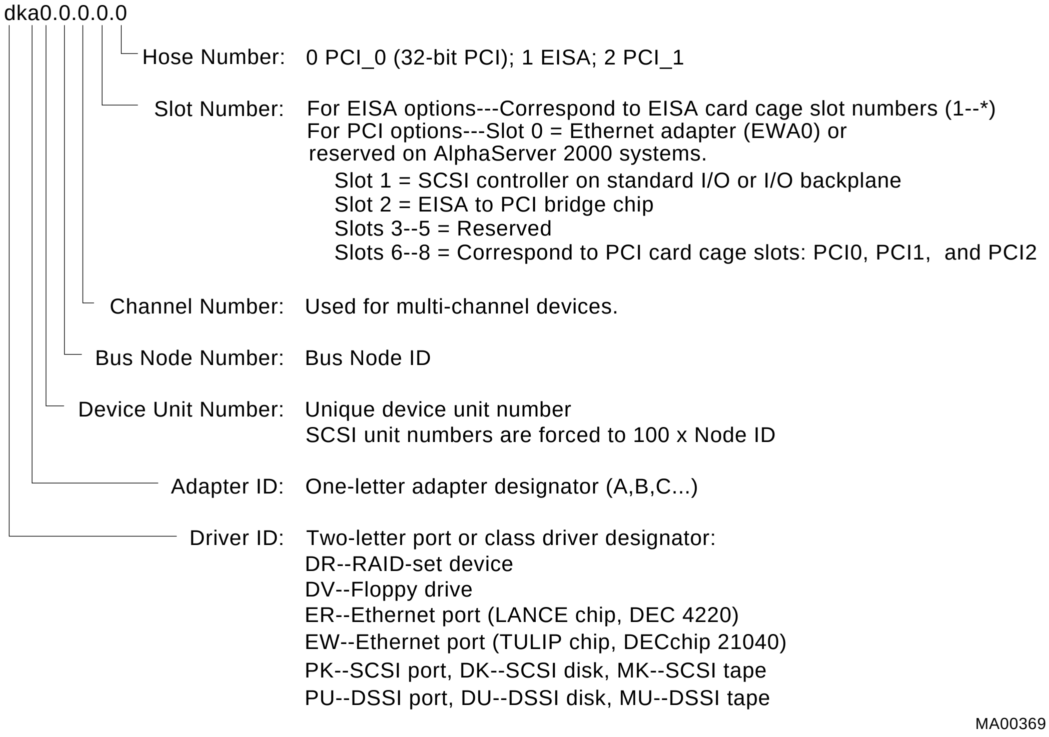
Example:

```
P00>>> show config
                        Digital Equipment Corporation
                           AlphaServer 2100 4/200

SRM Console V3.8-49             VMS PALcode X5.48-64, OSF PALcode X1.35-42

Component       Status          Module ID
CPU 0              P            B2020-AA DECchip (tm) 21064-3
Memory 0           P            B2021-BA  64 MB
I/O                             B2110-AA
                                dva0.0.0.0.1            RX26

 Slot   Option                  Hose 0, Bus 0, PCI
   0    DECchip 21040-AA        ewa0.0.0.0.0            08-00-2B-E2-56-2A
   1    NCR 53C810              pka0.7.0.1.0            SCSI Bus ID 7
                                dka0.0.0.1.0            RZ26
   2    Intel 82375EB                                   Bridge to Hose 1, EISA
   6    DECchip 21050-AA    Bridge to Bus 1, PCI


 Slot   Option                  Hose 0, Bus 1, PCI
   0    DECchip 7407

 Slot   Option                  Hose 1, Bus 0, EISA
 Slot   Option                  Hose 2, Bus 0, PCI
P00>>>
```

5.1.3.2 show device

The `show device` command displays the devices and controllers in the system. The device name convention is shown in Figure 5–2.

Figure 5–2 Device Name Convention


dka0.0.0.0.0
Hose Number: 0 PCI_0 (32-bit PCI); 1 EISA; 2 PCI_1
Slot Number: For EISA options---Correspond to EISA card cage slot numbers (1--*)
For PCI options---Slot 0 = Ethernet adapter (EWA0) or
reserved on AlphaServer 2000 systems.
Slot 1 = SCSI controller on standard I/O or I/O backplane
Slot 2 = EISA to PCI bridge chip
Slots 3--5 = Reserved
Slots 6--8 = Correspond to PCI card cage slots: PCI0, PCI1, and PCI2
Channel Number: Used for multi-channel devices.
Bus Node Number: Bus Node ID
Device Unit Number: Unique device unit number
SCSI unit numbers are forced to 100 x Node ID
Adapter ID: One-letter adapter designator (A,B,C...)
Driver ID: Two-letter port or class driver designator:
DR--RAID-set device
DV--Floppy drive
ER--Ethernet port (LANCE chip, DEC 4220)
EW--Ethernet port (TULIP chip, DECchip 21040)
PK--SCSI port, DK--SCSI disk, MK--SCSI tape
PU--DSSI port, DU--DSSI disk, MU--DSSI tape
MA00369

Synopsis:

show device [device_name]

Arguments:

[device_name]	The device name or device abbreviation. When abbreviations or wildcards are used, all devices that match the type are displayed.

Examples:

```
P00>>> show device
     ❶                        ❷       ❸                     ❹    ❺
dka0.0.0.1.0               DKA0                          RZ25   T392
dka300.3.0.1.0             DKA300                        RZ28   D41C
dka500.5.0.1.0             DKA500                        RRD43  0064
dva0.0.0.0.1               DVA0                          RX26
ewa0.0.0.0.0               EWA0              08-00-2B-3B-42-FD
pka0.7.0.1.0               PKA0                  SCSI Bus ID 7
P00>>>
```

❶ Console device name

❷ Abbreviated device name

❸ Node name (alphanumeric, up to 6 characters)

❹ Device type

❺ Firmware version (if known)

5.1.3.3 show memory

The `show memory` command displays information for each memory module in the system.

Synopsis:

`show memory`

Examples:

```
P00>>> show memory
  ❶      ❷       ❸          ❹           ❺          ❻
Module   Size   Base Addr   Intlv Mode  Intlv Unit  Status
------   -----  ---------   ----------  ----------  ------
  0       64MB  00000000      1-Way         0       Passed

Total Bad Pages 0 ❼
P00>>>
```

❶ Module slot number

❷ Size of memory module

❸ Base or starting address of memory module

❹ Interleave mode—number of modules interleaved (1–4-way interleaving)

❺ Interleave unit number

❻ Status (passed, failed, or not configured)

❼ Number of bad pages in memory (8 KB/page)

5.1.3.4 Setting and Showing Environment Variables

The environment variables described in Table 5–1 are typically set when you are configuring a system.

Synopsis:

set [-default] [-integer] -[string] envar value

Note

> Whenever you use the `set` command to reset an environment variable, you must initialize the system to put the new setting into effect. You initialize the system by entering the `init` command or pressing the Reset button.

show envar

Arguments:

envar The name of the environment variable to be modified.

value The value that is assigned to the environment variable. This may be an ASCII string.

Options:

-default Restores variable to its default value.

-integer Creates variable as an integer.

-string Creates variable as a string (default).

Examples:

```
P00>>> set bootdef_dev eza0
P00>>> show bootdef_dev
eza0
P00>>> show auto_action
boot
P00>>> set boot_osflags 0,1
P00>>>
```

Table 5–1 Environment Variables Set During System Configuration

Variable	Attributes	Function
auto_action	NV,W	The action the console should take following an error halt or powerfail. Defined values are: BOOT—Attempt bootstrap. HALT—Halt, enter console I/O mode. RESTART—Attempt restart. If restart fails, try boot. No other values are accepted. Other values result in an error message and variable remains unchanged.
bootdef_dev	NV	The device or device list from which booting is to be attempted, when no path is specified on the command line. Set at factory to disk with Factory Installed Software; otherwise null.

Key to variable attributes:

NV —- Nonvolatile. The last value saved by system software or set by console commands is preserved across system initializations, cold bootstraps, and long power outages.
W —- Warm nonvolatile. The last value set by system software is preserved across warm bootstraps and restarts.

(continued on next page)

Table 5–1 (Cont.) Environment Variables Set During System Configuration

Variable	Attributes	Function
boot_file	NV,W	The default file name used for the primary bootstrap when no file name is specified by the `boot` command. The default value when the system is shipped is NULL.
boot_osflags	NV,W	Default additional parameters to be passed to system software during booting if none are specified by the `boot` command. **OpenVMS:** On the OpenVMS AXP operating system, these additional parameters are the root number and boot flags. The default value when the system is shipped is NULL. **Digital UNIX:** The following parameters are used with the Digital UNIX operating system: a Autoboot. Boots /vmunix from bootdef_dev, goes to multiuser mode. Use this for a system that should come up automatically after a power failure. s Stop in single-user mode. Boots /vmunix to single-user mode and stops at the # (root) prompt. i Interactive boot. Request the name of the image to boot from the specified boot device. Other flags, such as -kdebug (to enable the kernel debugger), may be entered using this option. D Full dump, implies “s” as well. By default, if Digital UNIX crashes, it completes a partial memory dump. Specifying “D” forces a full dump at system crash. Common settings are a, autoboot; and Da, autoboot; but create full dumps if the system crashes.

Key to variable attributes:

NV —- Nonvolatile. The last value saved by system software or set by console commands is preserved across system initializations, cold bootstraps, and long power outages.
W —- Warm nonvolatile. The last value set by system software is preserved across warm bootstraps and restarts.

(continued on next page)

Table 5–1 (Cont.) Environment Variables Set During System Configuration

Variable	Attributes	Function
console	NV	Sets the device on which power-up output is displayed. GRAPHICS—Sets the power-up output to be displayed at a graphics terminal or device connected to the VGA module at the rear of the system. SERIAL—Sets the power-up output to be displayed on the device that is connected to the COM1 port at the rear of the system.
ew*0_mode	NV	Sets the Ethernet controller to the default Ethernet device type. aui—Sets the default Ethernet device to AUI. twisted—Sets the default Ethernet device to 10BASE-T (twisted-pair). auto—Reads the device connected to the Ethernet port and sets the default to the appropriate Ethernet device type.
er*0_protocols, ew*0_protocols	NV,W	Determines which network protocols are enabled for booting and other functions. “mop”—Sets the network protocol to mop: the setting typically used for systems using the OpenVMS operating system. “bootp”—Sets the network protocol to bootp: the setting typically used for systems using the Digital UNIX operating system. “bootp,mop”—When the settings are used in a list, the mop protocol is attempted first, followed by bootp.

Key to variable attributes:

NV —- Nonvolatile. The last value saved by system software or set by console commands is preserved across system initializations, cold bootstraps, and long power outages.
W —- Warm nonvolatile. The last value set by system software is preserved across warm bootstraps and restarts.

(continued on next page)

Table 5–1 (Cont.) Environment Variables Set During System Configuration

Variable	Attributes	Function
ocp_text	NV	Allows you to create an OCP message that displays when power-up diagnostics are completed. The default value is the CPU speed. Enter a message of up to 16 characters. Reset the system or enter the `init` command after setting ocp_text to activate new message.
os_type	NV	Sets the default operating system. "vms" or "osf"—Sets system to boot the SRM firmware. "nt"—Sets system to boot the ARC firmware.
pci_parity	NV	Disables or enables parity checking on the PCI bus. "on"—Enables parity checking for all devices on the PCI bus. "off"—Disables parity checking for all devices on the PCI bus.
pk*0_fast	NV	Enables fast SCSI devices on a SCSI controller to perform in standard or fast mode. 0—Sets the default speed for devices on the controller to standard SCSI. If a controller is set to standard SCSI mode, both standard and Fast SCSI devices will perform in standard mode. 1—Sets the default speed for devices on the controller to Fast SCSI mode. Devices on a controller that connect to both standard and Fast SCSI devices will automatically perform at the appropriate rate for the device, either fast or standard mode.

Key to variable attributes:

NV —- Nonvolatile. The last value saved by system software or set by console commands is preserved across system initializations, cold bootstraps, and long power outages.
W —- Warm nonvolatile. The last value set by system software is preserved across warm bootstraps and restarts.

(continued on next page)

Table 5–1 (Cont.) Environment Variables Set During System Configuration

Variable	Attributes	Function
pk*0_host_id	NV	Sets the controller host bus node ID to a value between 0 and 7. 0–7—Assigns bus node ID for specified host adapter.

Key to variable attributes:

NV —- Nonvolatile. The last value saved by system software or set by console commands is preserved across system initializations, cold bootstraps, and long power outages.
W —- Warm nonvolatile. The last value set by system software is preserved across warm bootstraps and restarts.

Note

Whenever you use the `set` command to reset an environment variable, you must initialize the system to put the new setting into effect. Initialize the system by entering the `init` command or pressing the Reset button.

5.2 System Bus Options

The system bus interconnects the CPUs, memory modules, and the optional PCI extended I/O module. Figure 5–3 and Figure 5–4 show the location of the system bus and card cages for Digital Alpha 2100 VME systems.

Figure 5–3 Card Cages and Bus Locations (Vertical-Mount)

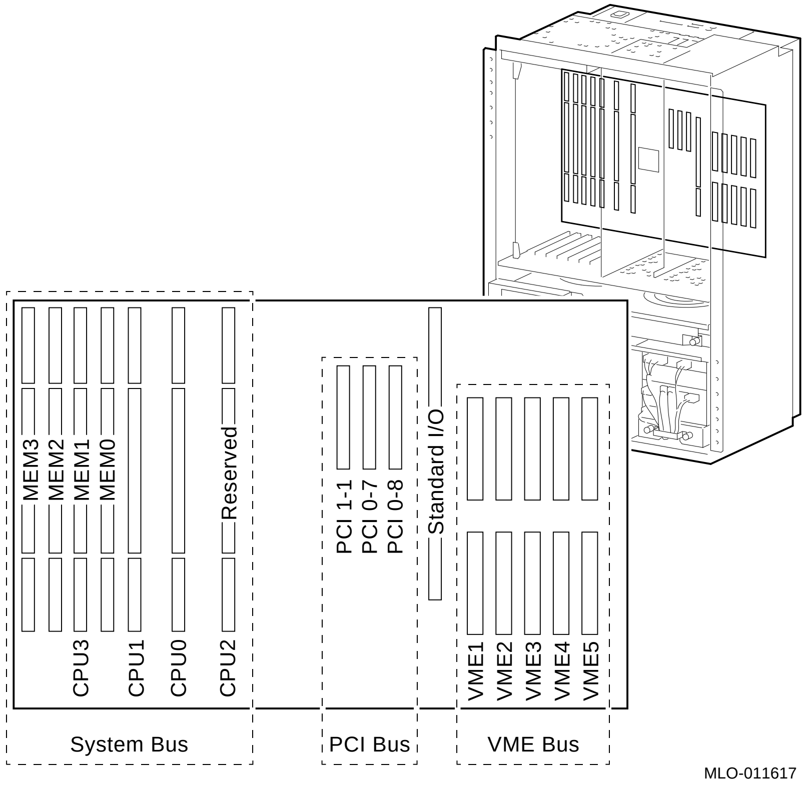
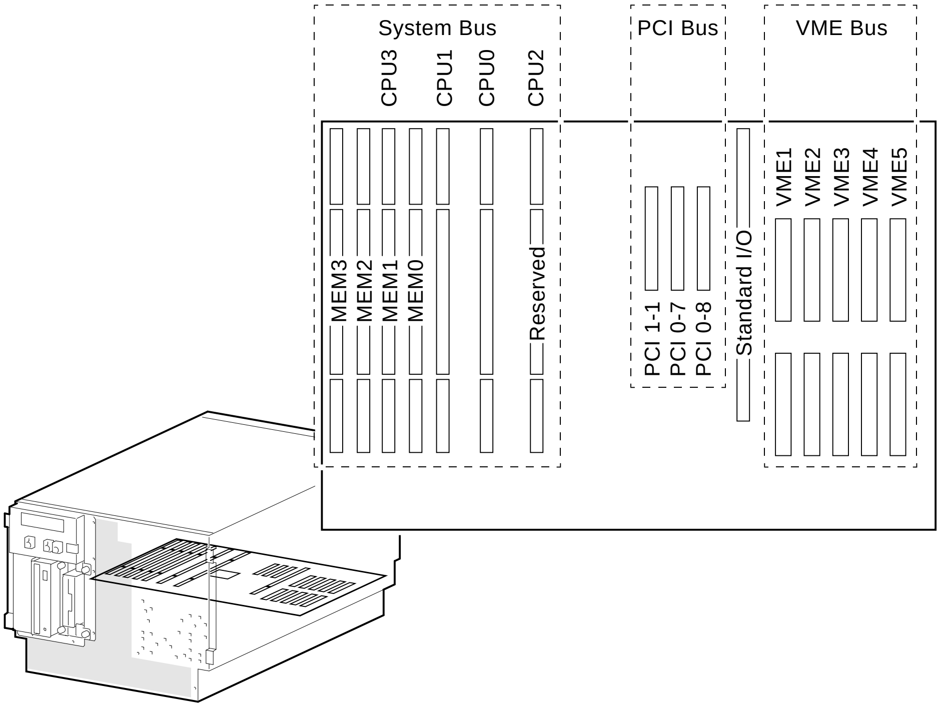


Figure 5–4 Card Cages and Bus Locations (Drawer-Mount)



MLO-011619

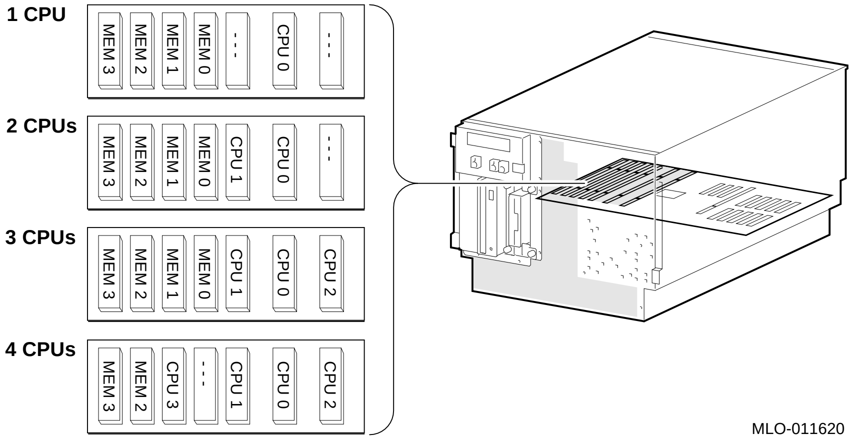
5.2.1 CPU Modules

Digital Alpha 2100 VME systems can support up to four CPUs in a symmetric multiprocessing (SMP) configuration.

- All systems must have a CPU module installed in system bus slot 2 (CPU 0).
- Systems with more than two CPUs displace memory module capacity as shown in Figure 5–5 and Figure 5–6.

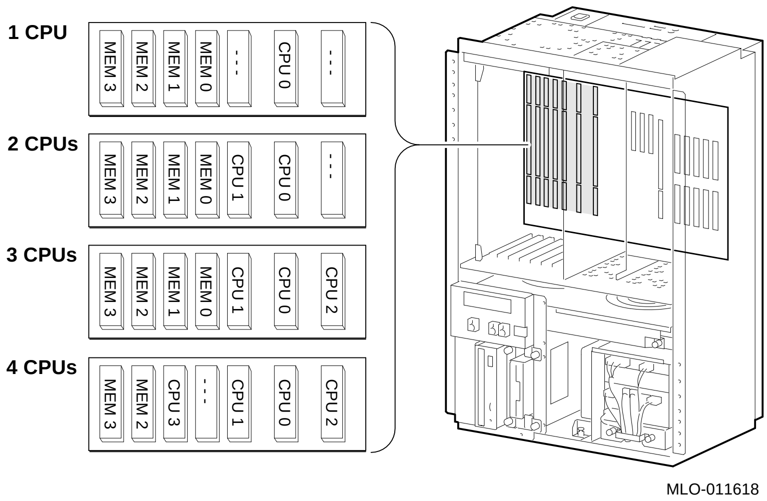
Warning: CPU and memory modules have parts that operate at high temperatures. Wait two minutes after power is removed before handling these modules.

Figure 5–5 System Bus Configurations According to Number of CPUs (Drawer-Mount)



1 CPU	MEM 3	MEM 2	MEM 1	MEM 0	- - -	CPU 0	- - -
2 CPUs	MEM 3	MEM 2	MEM 1	MEM 0	CPU 1	CPU 0	- - -
3 CPUs	MEM 3	MEM 2	MEM 1	MEM 0	CPU 1	CPU 0	CPU 2
4 CPUs	MEM 3	MEM 2	CPU 3	- - -	CPU 1	CPU 0	CPU 2

Figure 5–6 System Bus Configurations According to Number of CPUs (Vertical-Mount)



1 CPU	MEM 3	MEM 2	MEM 1	MEM 0	- - -	CPU 0	- - -
2 CPUs	MEM 3	MEM 2	MEM 1	MEM 0	CPU 1	CPU 0	- - -
3 CPUs	MEM 3	MEM 2	MEM 1	MEM 0	CPU 1	CPU 0	CPU 2
4 CPUs	MEM 3	MEM 2	CPU 3	- - -	CPU 1	CPU 0	CPU 2

5.2.2 Memory Modules

Digital Alpha VME 2100 systems can support up to four memory modules (for a maximum memory capacity of 2 GB). A minimum of one memory module is required.

Memory is available in three variations:

- MS450–BA (B2021–BA) 64-MB memory
- MS450–CA (B2021–CA) 128-MB memory
- MS451–CA (B2022–CA) 512-MB memory

5.3 Standard I/O Module

The standard I/O module provides a standard set of I/O functions. The standard I/O module resides in a dedicated slot (I/O) in the PCI bus card cage.

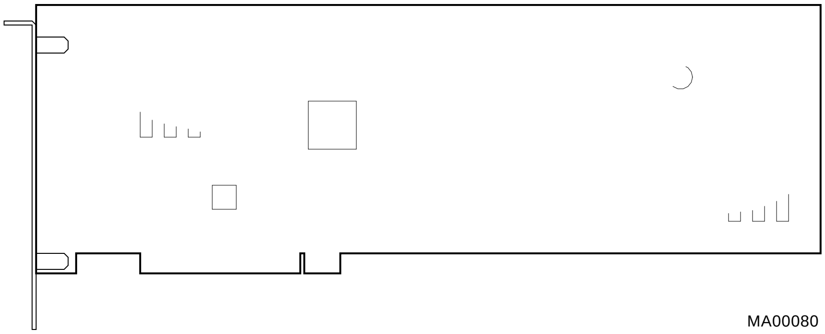
The standard I/O module and I/O backplane provide:

- A Fast SCSI-2 controller chip that supports up to seven drives.
- The firmware console subsystem on 1 MB of Flash ROM.
- An Ethernet controller with AUI and twisted-pair connectors.
- A floppy drive controller.
- Two serial ports with full modem control and the parallel port.
- The keyboard and mouse interface.
- The speaker interface.
- PCI-to-EISA bridge chip set.
- Time-of-year (TOY) clock

5.4 PCI Bus Options

PCI (Peripheral Component Interconnect) is an industry-standard expansion I/O bus that is the preferred bus for high-performance I/O options. Up to three 32-bit PCI options can reside in the PCI portion of the card cage. A PCI board is shown in Figure 5–7.

Figure 5–7 PCI Board



Install PCI boards according to the instructions supplied with the option. PCI boards require no additional configuration procedures; the system automatically recognizes the boards and assigns the appropriate system resources.

Warning: For protection against fire, only modules with current-limited outputs should be used.

5.5 VME Bus Options

VME (Versa Module Eurocard) is an industry-standard expansion I/O bus. Digital Alpha 2100 VME systems support up to five VME modules (6U VME form factor).

Digital Alpha 2100 VME systems ship with BG0-3 and IACK jumpers installed on the front of the motherboard (Figure 5–8). These jumpers must be removed for each VME module that is installed (except in the case of slave modules).

Note

VME jumpers can be installed on either side of the motherboard, front or rear. If you choose to use the rear of the motherboard, all the jumpers should be installed from the rear; if you choose to use the front of the motherboard, all the jumpers should be installed from the front.

If a VME slot is not filled with a VME module or is filled with a slave module that does not connect the “IN” and “OUT” daisy chains and if other modules are installed in other slots, then the jumpers must be installed at that slot in order to pass through the bus grant and interrupt acknowledge daisy-chain signals.

Use the power configuration worksheet provided in Figure 5–9 to configure the VME bus options.

Figure 5–8 VME Backplane Jumpers

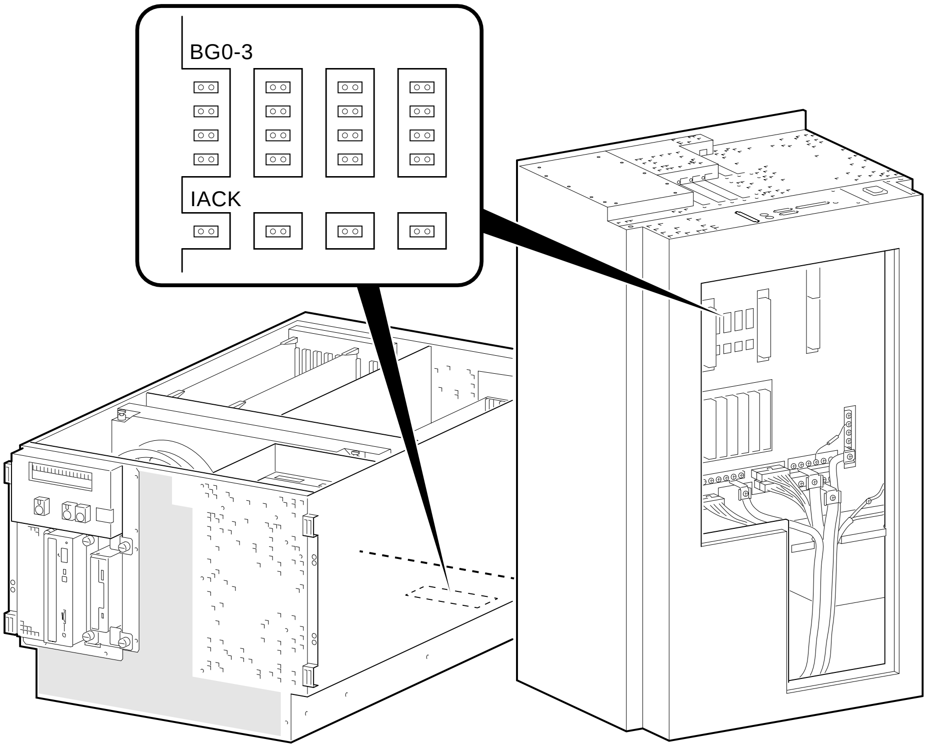
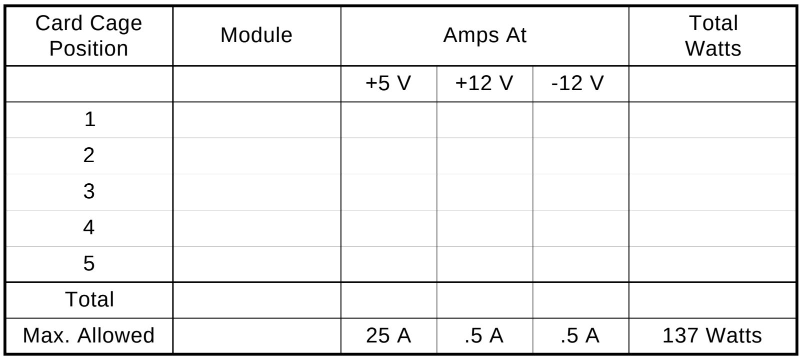


Figure 5–9 VME Bus Power Configuration Worksheet



Card Cage Position	Module	Amps At			Total Watts
		+5 V	+12 V	-12 V	
1					
2					
3					
4					
5					
Total					
Max. Allowed		25 A	.5 A	.5 A	137 Watts

MLO-011661

5.5.1 Installing a Typical 6U VME Module

To install a VME options, complete the following steps:

1. Determine which VME slot (1 through 5) that 6U module will be installed.
2. Remove or install backplane jumpers.
3. Using antistatic protection, remove the module from its package and check for damaged or loose components. Follow the instructions provided with the module for installation and configuration procedures.
4. Align the module with the card guides and insert the module into the slot.
5. When resistance is felt, continue to push the module into the slot until the connectors are fully seated.
6. Secure the module in the card cage by tightening the screws on the module's handle.
7. Connect cables according to instructions provided with the module.

5.5.2 VME Backplane Connector Pin Assignments

Table 5–2 and Table 5–3 provide the pin assignments for the P1 and P2 VME backplane connectors. These connectors consist of three rows of pins labeled (a), (b), and (c).

Table 5–2 P1 Pin Assignments

Pin Number	(a) Signal Mnemonic	(b) Signal Mnemonic	(c) Signal Mnemonic
1	D00	BBSY*	D08
2	D01	BCLR*	D09
3	D02	ACFAIL*	D10
4	D03	BG0IN*	D11
5	D04	BG0OUT*	D12
6	D05	BGIN*	D13
7	D06	BG1OUT*	D14
8	D07	BG2IN*	D15
9	GND	BG2OUT*	GND
10	SYSCLK	BG3IN*	SYSFAIL*
11	GND	BG3OUT*	BERR*
12	DS1*	BR0*	SYSRESET*
13	DS0*	BR1*	LWORD*
14	WRITE*	BR2*	AM5
15	GND	BR3*	A23
16	DTACK*	AM0	A22
17	GND	AM1	A21
18	AS*	AM2	A20
19	GND	AM3	A19
20	IACK*	GND	A18
21	IACKIN*	SERCLK	A17
22	IACKOUT	SERDAT*	D16
23	AM4	GND	A15
24	A07	IRQ7*	A14

(continued on next page)

Table 5–2 (Cont.) P1 Pin Assignments

Pin Number	(a) Signal Mnemonic	(b) Signal Mnemonic	(c) Signal Mnemonic
25	A06	IRQ6*	A13
26	A05	IRQ5*	A12
27	A04	IRQ4*	A11
28	A03	IRQ3*	A10
29	A02	IRQ2*	A09
30	A01	IRQ1*	A08
31	-12 V	+5V STDBY	+12V
32	+5V	+5V	+5V

Table 5–3 P2 Pin Assignments

Pin Number	(a) Signal Mnemonic	(b) Signal Mnemonic	(c) Signal Mnemonic
1	User-Defined	+5V	User-Defined
2	User-Defined	GND	User-Defined
3	User-Defined	RESERVED	User-Defined
4	User-Defined	A24	User-Defined
5	User-Defined	A25	User-Defined
6	User-Defined	A26	User-Defined
7	User-Defined	A27	User-Defined
8	User-Defined	A28	User-Defined
9	User-Defined	A29	User-Defined
10	User-Defined	A30	User-Defined
11	User-Defined	A31	User-Defined
12	User-Defined	GND	User-Defined

(continued on next page)

Table 5–3 (Cont.) P2 Pin Assignments

Pin Number	(a) Signal Mnemonic	(b) Signal Mnemonic	(c) Signal Mnemonic
13	User-Defined	+5V	User-Defined
14	User-Defined	D16	User-Defined
15	User-Defined	D17	User-Defined
16	User-Defined	D18	User-Defined
17	User-Defined	D19	User-Defined
18	User-Defined	D20	User-Defined
19	User-Defined	D21	User-Defined
20	User-Defined	D22	User-Defined
21	User-Defined	D23	User-Defined
22	User-Defined	GND	User-Defined
23	User-Defined	D24	User-Defined
24	User-Defined	D25	User-Defined
25	User-Defined	D26	User-Defined
26	User-Defined	D27	User-Defined
27	User-Defined	D28	User-Defined
28	User-Defined	D29	User-Defined
29	User-Defined	D30	User-Defined
30	User-Defined	D31	User-Defined
31	User-Defined	GND	User-Defined
32	User-Defined	+5V	User-Defined

5.6 SCSI Buses

A Fast SCSI-2 adapter on the standard I/O module provides a single-ended SCSI bus for Digital Alpha 2100 VME systems.

All tabletop or rackmounted SCSI-2 devices are supported via VME- or PCI-based SCSI adapters. Use the following rules to determine if a device can be used on your system:

- The device must be supported by the operating system. Consult the software product description or hardware vendor.
- No more than seven devices can be on any one SCSI-2 controller, and each must have a unique SCSI ID.
- The entire SCSI bus length, from terminator to terminator, must not exceed 6 meters for single-ended SCSI-2 at 5 MB/sec, or 3 meters for single-ended SCSI-2 at 10 MB/sec.
  - For BA742 rackmount enclosures, the internal cabling for the removable media and internal disk-drives is 2 meters; therefore, the maximum length for external expansion is 4 meters.

5.6.1 Internal SCSI Bus

The Fast SCSI-2 adapter on the standard I/O module supports the internal SCSI drives:

One or two hard disk drives and up to two 5.25-inch, half-height devices

This bus can be extended to a rack-mounted StorageWorks shelf or to an external expander to support up to seven drives.

5.6.2 Installing Removable Media Devices

Figure 5–10 shows how to install 5.25-in. half-height devices in the removable-media compartment. Use the screws (M3 x 6 mm, flathead) supplied in the accessories kit to mount the drives.

Be sure that you set the device's node ID so that there are no duplicate node IDs, as each device must have a unique node ID. Nodes 0–6 are available for drives, and node 7 is reserved for the host adapter. For information on device switch settings, refer to the documentation supplied with the device.

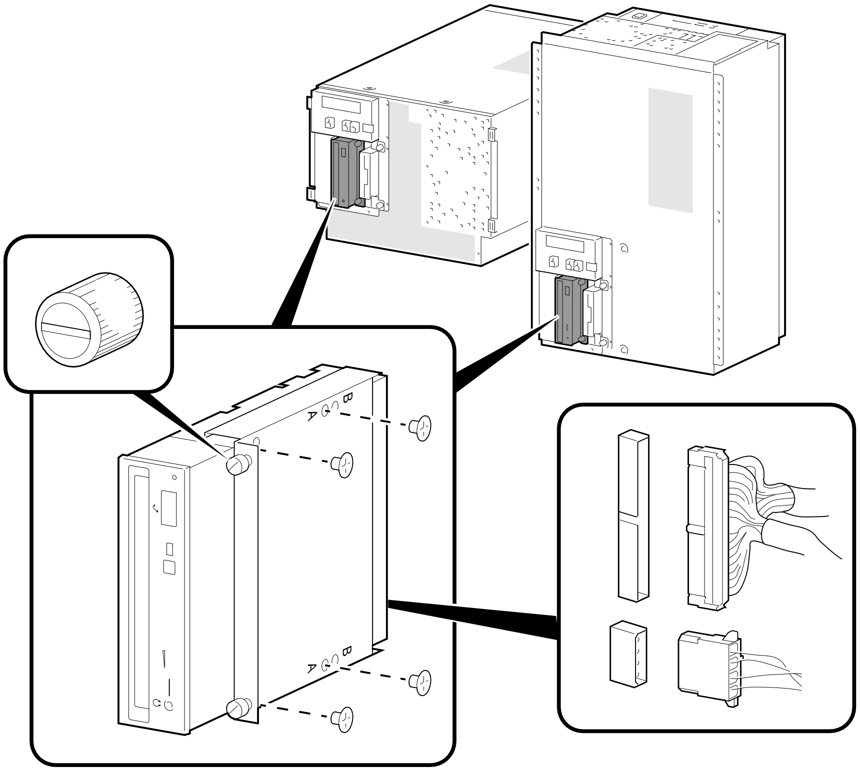
Note

RRDnn and TLZ0n drives use the set of bracket holes marked "A" in Figure 5–10.

The TZK11 drive uses the set of bracket holes marked "B" in Figure 5–10.

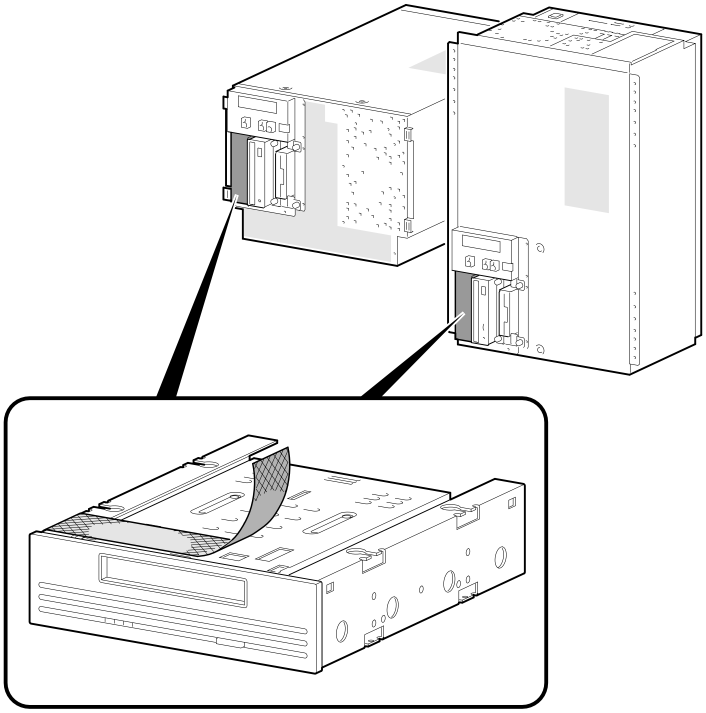
A plastic strip (70-32518-01) must be attached to a TLZ0n tape drive when the tape drive is installed as the left-most removable-media device in a Digital Alpha VME 2100 system with no front bezel (Figure 5–11).

Figure 5–10 Installing Removable Media



MLO-011610a

Figure 5–11 Plastic Strip for TLZ0n Tape Drives



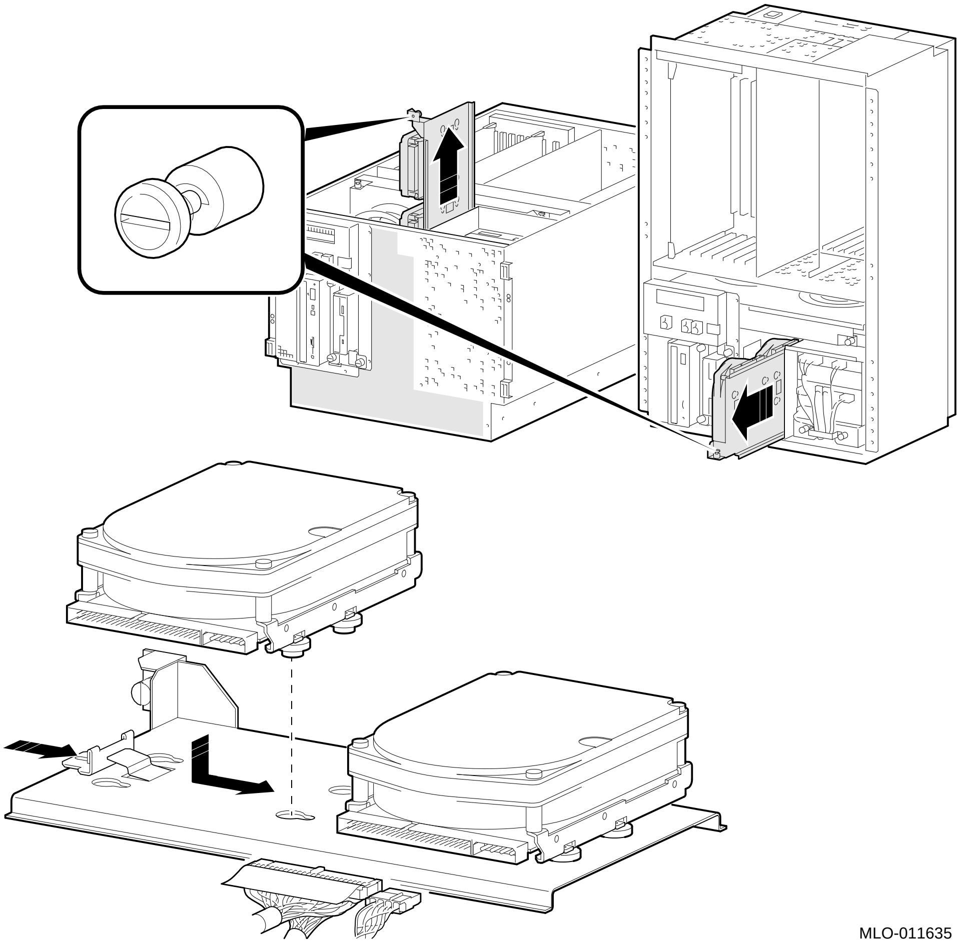
5.6.3 Installing Fixed-Disks

To install a fixed-disk drive:

1. Mount four rubber grommets provided in the accessories kit to the drive.
2. Install drive as shown in Figure 5–12.

Be sure that you set the device's node ID so that there are no duplicate node IDs, as each device must have a unique node ID. Nodes 0–6 are available for drives, and node 7 is reserved for the host adapter. For information on device switch settings, refer to the documentation supplied with the device.

Figure 5–12 Installing a Fixed-Disk Drive



5.7 Console Port Configurations

Power-up information is typically displayed on the system's console terminal. The console terminal may be either a graphics terminal or a serial terminal (connected through the COM1 serial port). The setting of the console environment variable determines where the system will display power-up output. Set this environment variable according to the console terminal that you are using.

Synopsis:

set console output_device

Arguments:

graphics	Displays the power-up output to a graphics terminal or device connected to the VGA module at the rear of the system.
serial	Displays the power-up output to a device connected to the COM1 port at the rear of the system.

Example:

```
P00>>> set console serial
P00>>>
```

VTxxx Console Terminal Setting for Running ECU

To run the EISA configuration utility (ECU) from the serial console port, the terminal needs to bet set for 8-bit controls, the keyboard needs to be set so that the tilde (~) key sends the escape (ESC) signal, and the console environment variable must be set to serial.

6

Digital Alpha VME 2100 (BA742 Enclosure) FRU Removal and Replacement

This chapter describes the field-replaceable unit (FRU) removal and replacement procedures for Digital Alpha VME 2100 systems, which use the BA742 enclosure.

- Section 6.1 lists the FRUs for Digital Alpha VME 2100 systems (BA742 enclosure).
- Section 6.2 provides the removal and replacement procedures for the FRUs.

6.1 Digital Alpha VME 2100 (BA742 Enclosure) FRUs

Table 6–1 lists the FRUs by part number and description and provides the reference to the figure or section that provides the removal/replacement procedure.

The BA742 can be used in two orientations:

- Vertical rackmount
- Drawer rackmount

Drawer Rackmount BA742

6–1 shows the locations of FRUs within the drawer-mounted BA742 enclosure.

Section 6.2.1 shows how to access the drawer-mounted FRUs.

Vertical Rackmount BA742

Figure 6–2 shows the locations of FRUs within the vertically-mounted BA742 enclosure.

Section 6.2.2 shows how to access the vertically-mounted FRUs.

Table 6–1 BA742 Enclosure FRUs

Part #	Description	Section
Cables		
17-04137-01	Floppy drive cable (34-pin)	Figure 6–9
17-04133-01	Multinode power distribution cable (4-pin)	Figure 6–10
17-04135-01	OCP module cable (10-pin)	Figure 6–11
17-00083-51	Power cord (external)	Figure 6–12
17-04175-01	Power supply control cable assembly	Figure 6–13 Figure 6–14
17-04156-02	Power supply +3.3V cable	Figure 6–15 Figure 6–16
17-04156-01	Power supply +3.3V return cable	Figure 6–15 Figure 6–16
17-04167-02	Power supply +5.0V cable	Figure 6–15 Figure 6–16
17-04167-01	Power supply +5.0V return cable	Figure 6–15 Figure 6–16
17-04158-01	Remote I/O cable (60-pin)	Figure 6–17
17-04136-01	SCSI multinode cable (50-pin)	Figure 6–18
17-04134-01	-12 V Converter to backplane cable	Figure 6–19
CPU Modules		
B2020-AA	KN450-AA CPU module	Section 6.2.4
B2024-AA	KN460-AA CPU module	Section 6.2.4
Fans		
12-23374-02	6.75-inch fans	Section 6.2.5 Section 6.2.6
54-22615-01	Fan speed control module	Section 6.2.7

(continued on next page)

Table 6–1 (Cont.) BA742 Enclosure FRUs

Part #	Description	Section
I/O Modules		
B2110-AA (54-23146-01)	KFE40 standard I/O	Section 6.2.8
54-23151-01	Remote I/O module	Section 6.2.9
Memory Modules		
B2021-BA	MS450-BA 64MB memory module	Section 6.2.10
B2021-CA	MS450-CA 128MB memory module	Section 6.2.10
B2022-CA	MS451-CA 512MB memory module	Section 6.2.10
Other Modules and Components		
54-22629-01	Motherboard	Section 6.2.11 Section 6.2.12
54-23180-03	OCP module	Section 6.2.13 Section 6.2.14
54-22631-01	PCI to VME daughter board	Section 6.2.15
30-44153-01	Power supply	Section 6.2.16
12-39309-03	Speaker	Section 6.2.17
54-22639-01	Voltage protection module (MOV)	Section 6.2.18
54-22649-01	-12 V Converter module	Section 6.2.19
90-11194-01	Key, 1/4-turn fastener	
12-37004-04	External SCSI terminator	
Removable Media		
RRDnn-AA	CD–ROM drives	Section 6.2.20
TLZnn-LG	Tape drives	Section 6.2.20
TZKnn-LG	Tape drives	Section 6.2.20
RXnn-AA	Floppy drive	Section 6.2.20

(continued on next page)

Table 6–1 (Cont.) BA742 Enclosure FRUs

Part #	Description	Section
Fixed Disk Drives		
RZnn -AA	Disk drive	Section 6.2.21

Figure 6–1 FRUs, Drawer-Mount

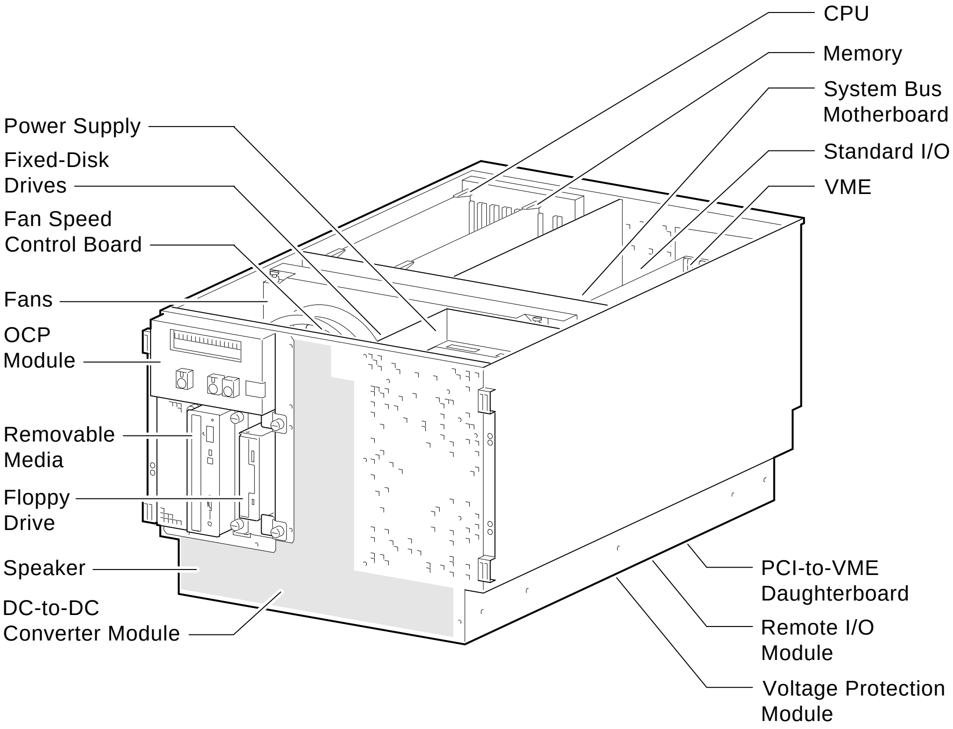
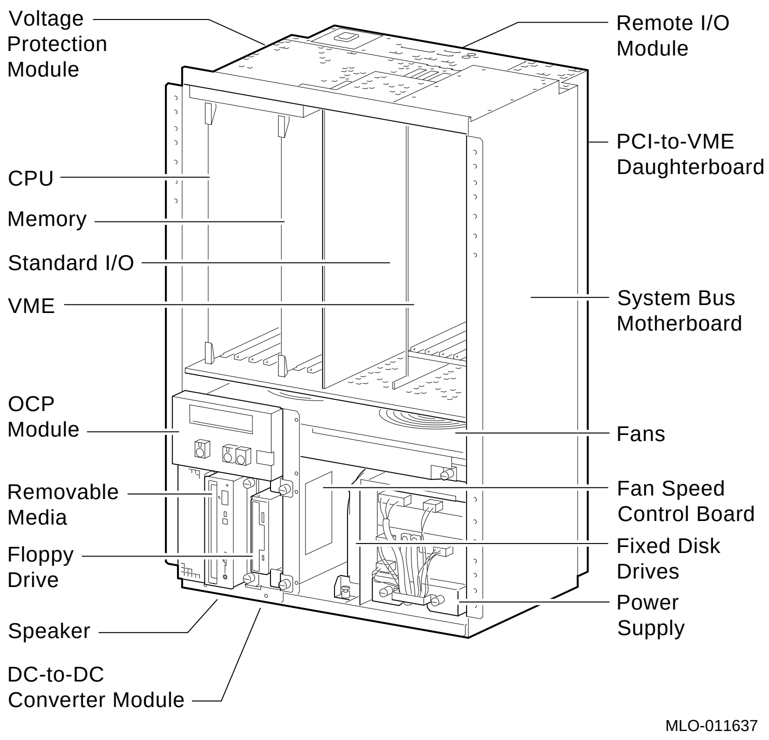


Figure 6–2 FRUs, Vertical-Mount



6.2 Removal and Replacement

This section describes the procedures for removing and replacing FRUs for Digital Alpha 2100 VME systems, which use the BA742 enclosure.

Warning: Before accessing enclosure compartments:

1. Perform orderly shutdown of the operating system.
2. Set the DC power switch on the operator control panel to off.
3. Remove power by unplugging the AC power cord from the power supply.

Caution

Static electricity can damage integrated circuits. Always use a grounded wrist strap (29-26246) and grounded work surface when working with internal parts of a computer system.

Unless otherwise specified, you can install an FRU by reversing the steps shown in the removal procedure.

6.2.1 Accessing Drawer-Mount Components

Warning

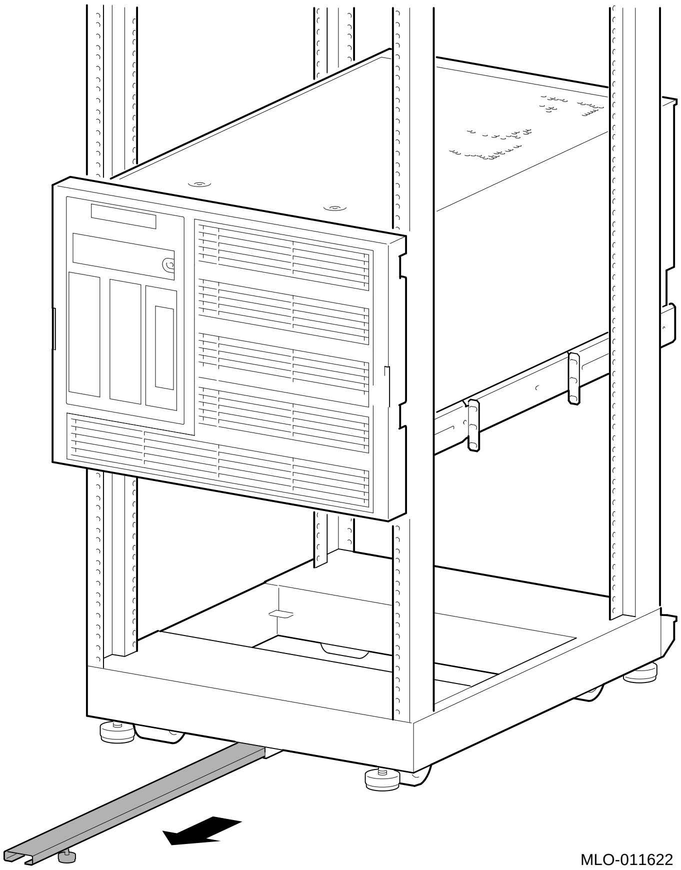
The system weighs 45.4 kg (100 lb). To prevent personal injury and equipment damage, ensure that only one system is extended out of the cabinet at any one time and that the cabinet is stabilized (as in Figure 6–3) before pulling the system out on its slides.

The adjustable leveling feet should be down and the cabinet's stabilizing bar fully extended before any component is extended out of the cabinet on slides.

Do not extend more than one slide assembly at a time, otherwise cabinet instability may result.

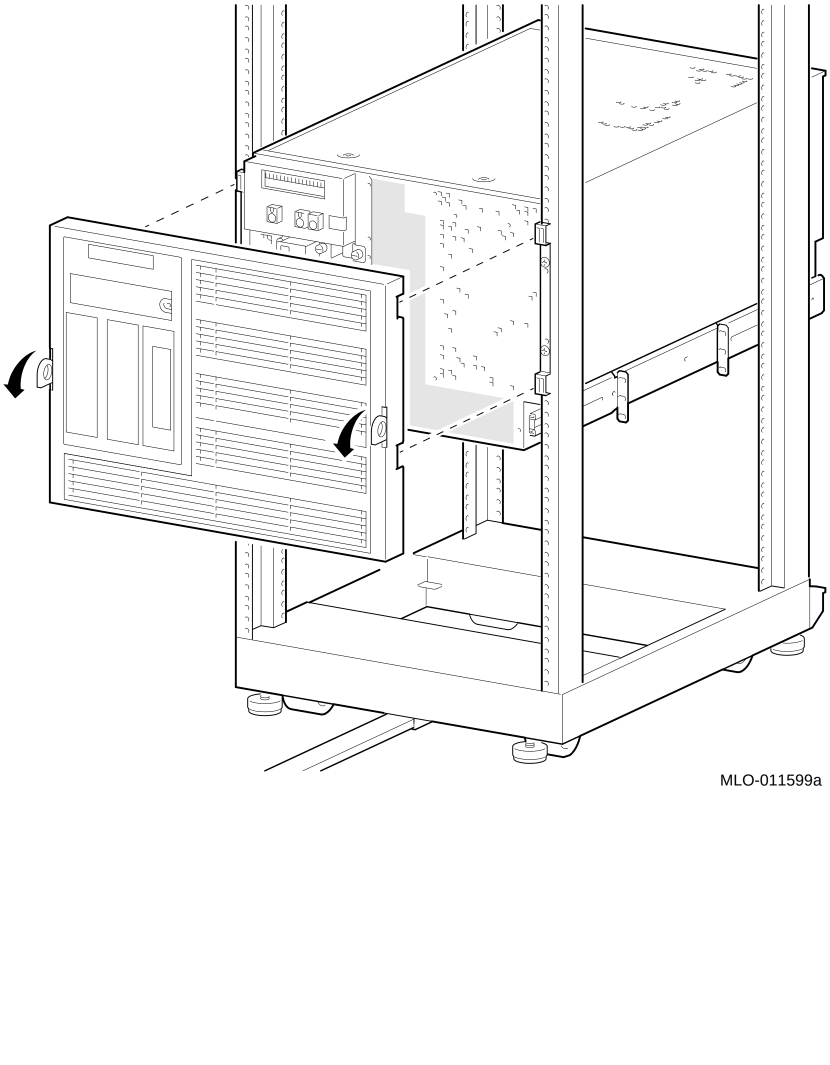
STEP 1: STABILIZE CABINET BEFORE SLIDING SYSTEM OUT.

Figure 6–3 Example of a Cabinet Stabilizer



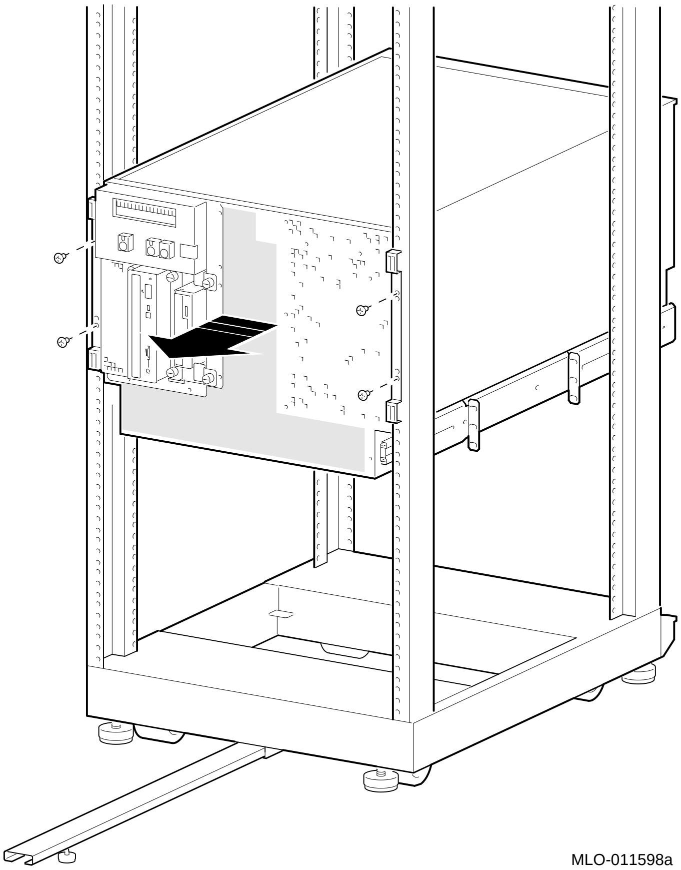
STEP 2: REMOVE FRONT PANEL.

Figure 6–4 Removing Front Panel



STEP 3: REMOVE SCREWS AND SLIDE SYSTEM OUT.

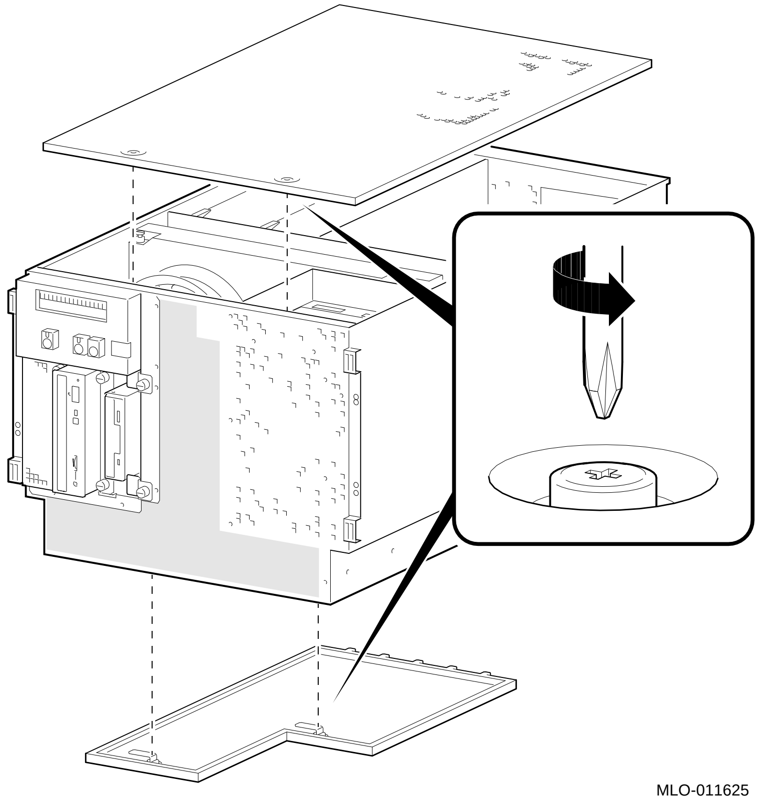
Figure 6–5 Sliding Out Rackmount System



STEP 4: REMOVE ENCLOSURE COVERS.

Unscrew the pawl latches until the covers release.

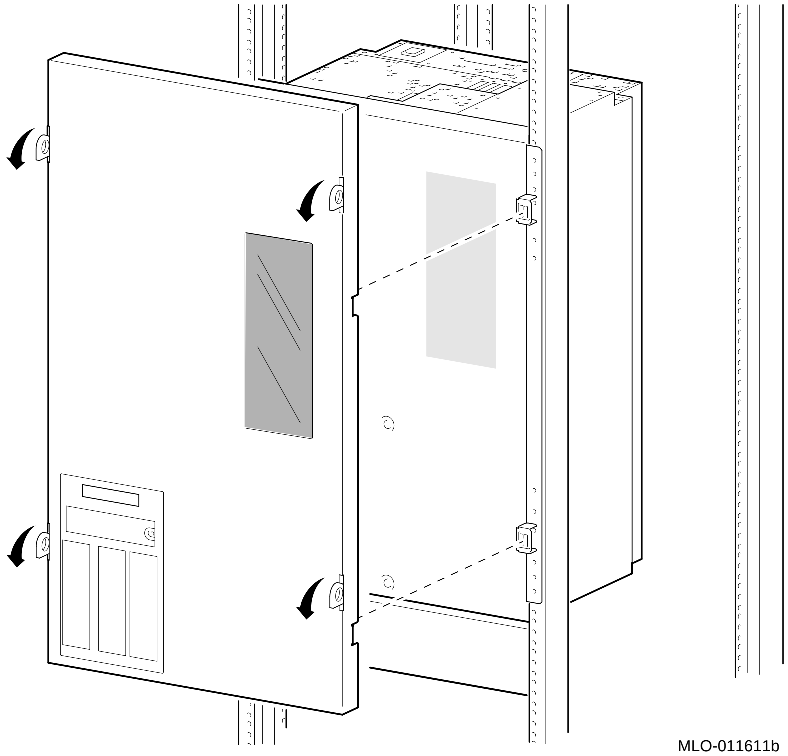
Figure 6–6 Removing Drawer-Mount Top and Bottom Covers



6.2.2 Accessing Vertical-Mount Components

STEP 1: REMOVE FRONT PANEL.

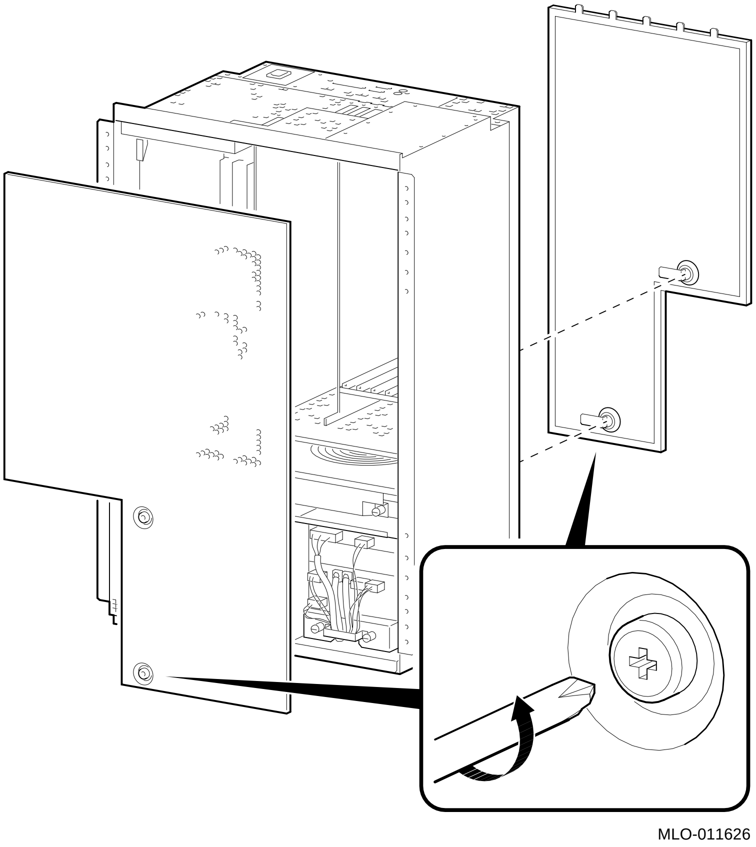
Figure 6–7 Removing Front Panel (Vertical Mount)



STEP 2: REMOVE VERTICAL-MOUNT FRONT AND REAR COVERS.

Unscrew the pawl latches until the covers release.

Figure 6–8 Removing Front and Rear Covers (Vertical Mount)



6.2.3 Cables

This section shows the routing for each cable in the system.

Figure 6–9 Floppy Drive Cable (34-pin)

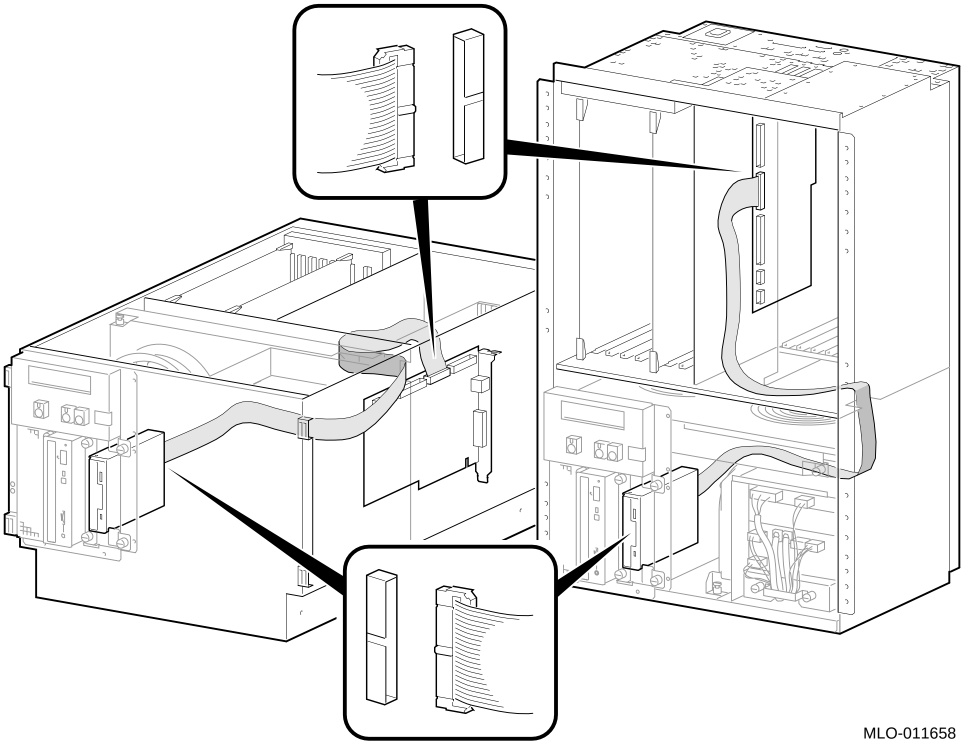


Figure 6–10 Multinode Power Distribution Cable (4-pin)

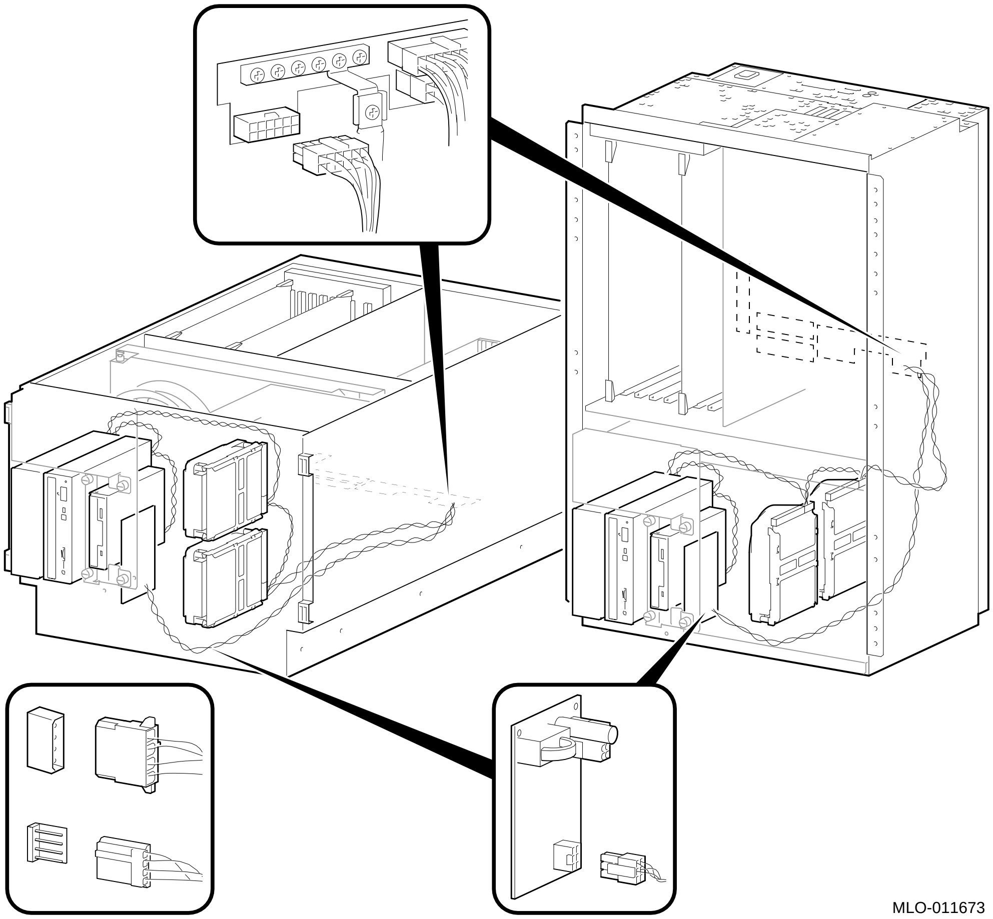


Figure 6–11 OCP Module Cable (10-pin)

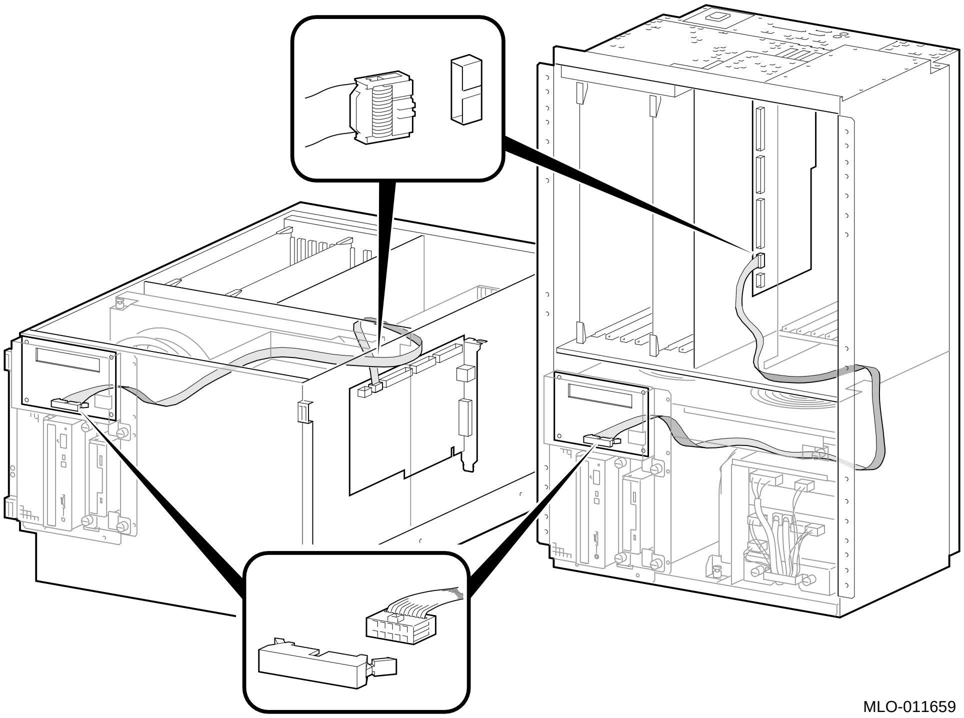


Figure 6–12 Power Cord

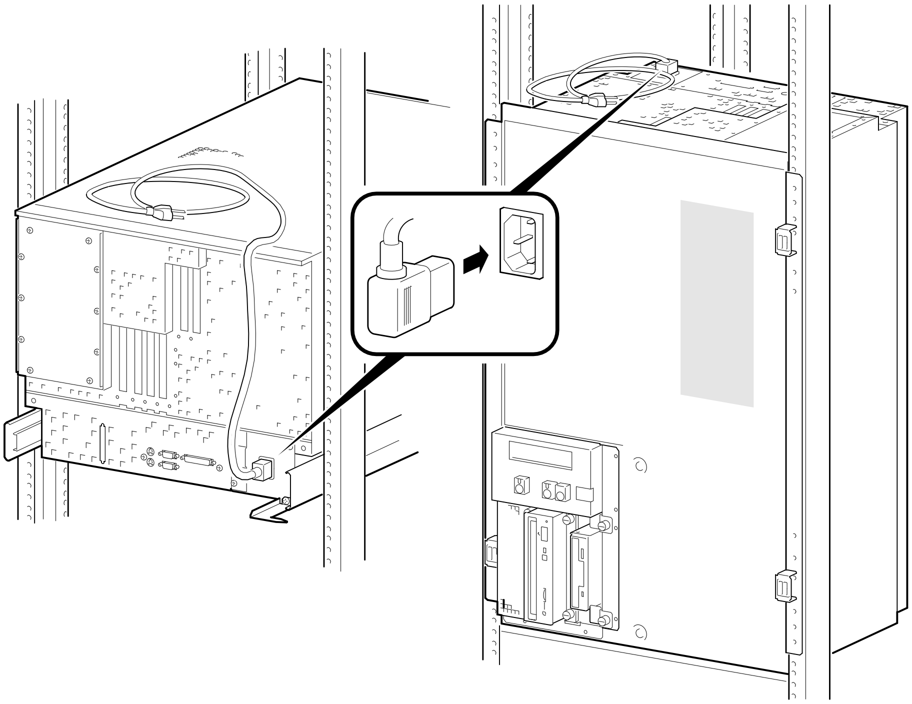
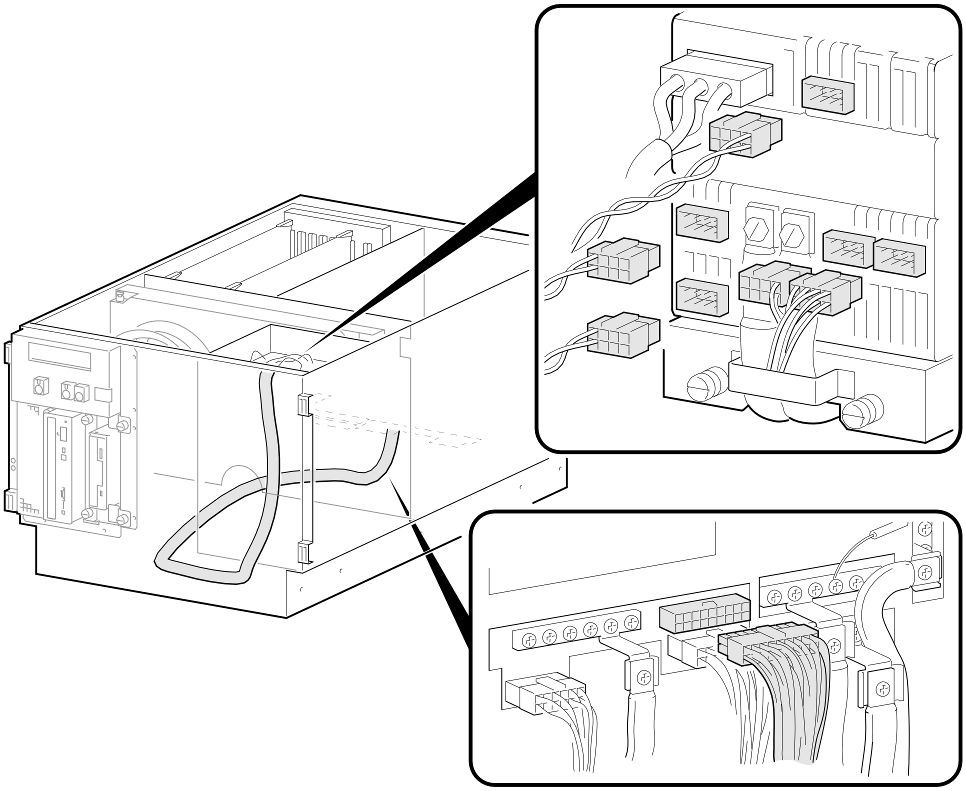


Table 6–2 lists the country-specific power cables.

Table 6–2 Power Cord Order Numbers

Country	Power Cord BN Number	Digital Number
U.S., Japan, Canada	Included	17-00083-51
Australia, New Zealand	BN19J-2E	17-00198-13
Central European (Aus, Bel, Fra, Ger, Fin, Hol, Nor, Swe, Por, Spa)	BN19D-2E	17-00199-22
U.K., Ireland	BN19B-2E	17-00209-12
Switzerland	BN04B-2E	17-00210-12
Denmark	BN19L-2E	17-00310-06
Italy	BN19N-2E	17-00364-17
India, South Africa	BN19T-2E	17-00456-15
Israel	BN18Y-2E	17-00457-15

Figure 6–13 Power Supply Control Cable Assembly (Drawer-Mount)



MLO-011663

Figure 6–14 Power Supply Control Cable Assembly (Vertical-Mount)

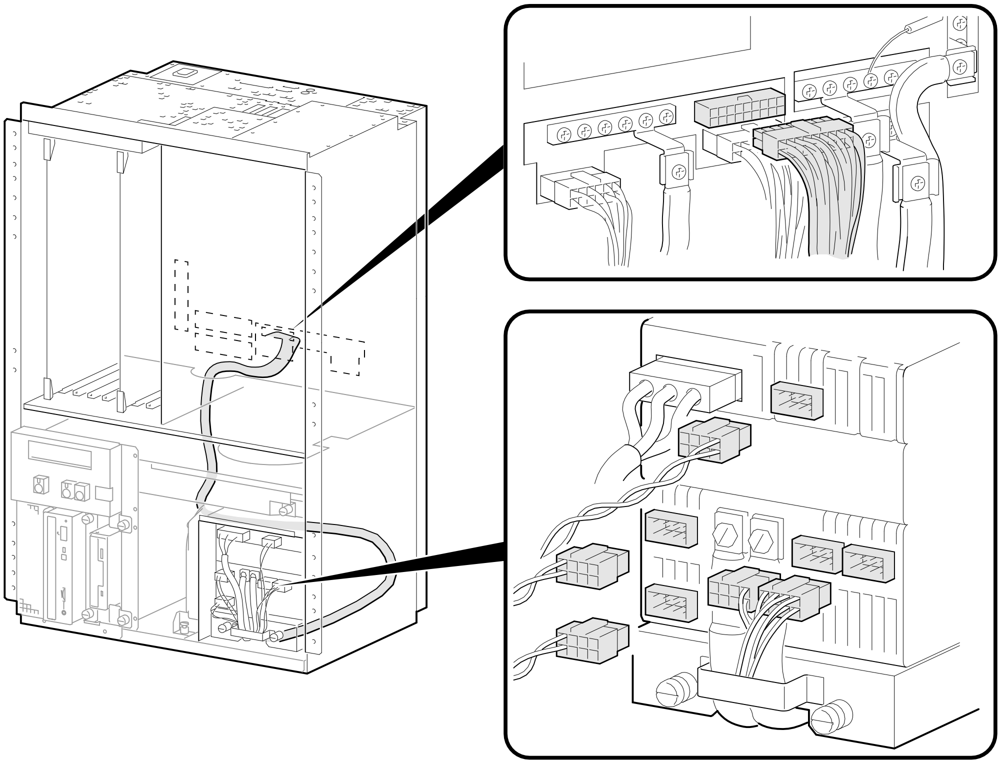


Figure 6–15 Power Supply +3.3V and +5.0V Cables (Drawer-Mount)

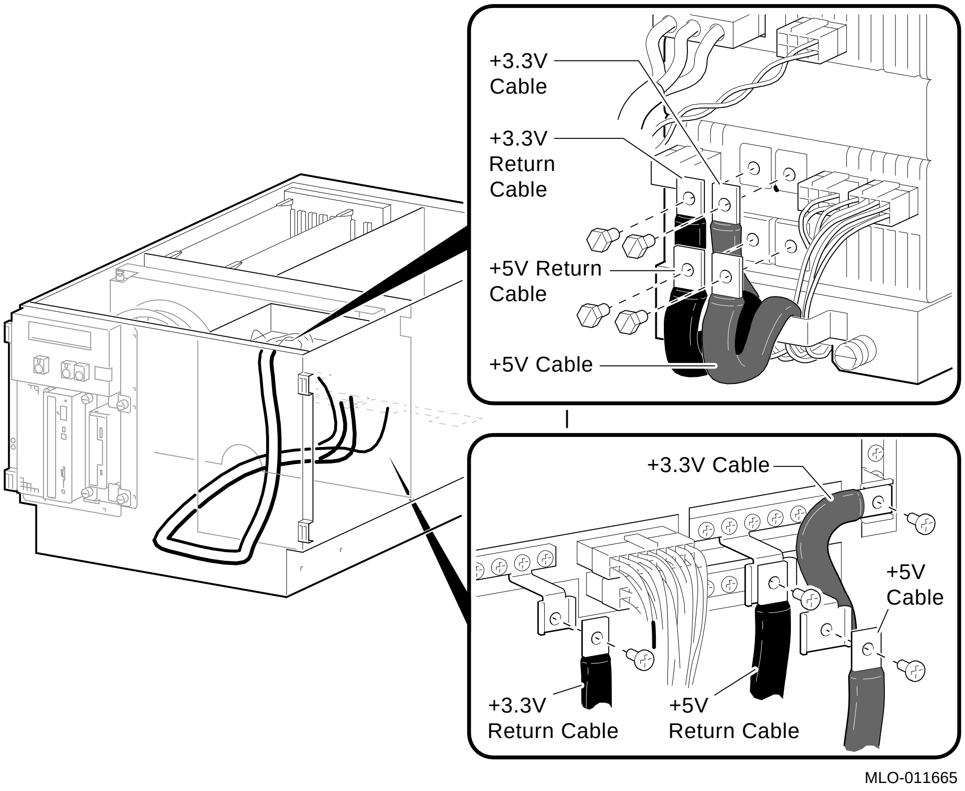


Figure 6–16 Power Supply +3.3V and +5.0V Cables (Vertical-Mount)

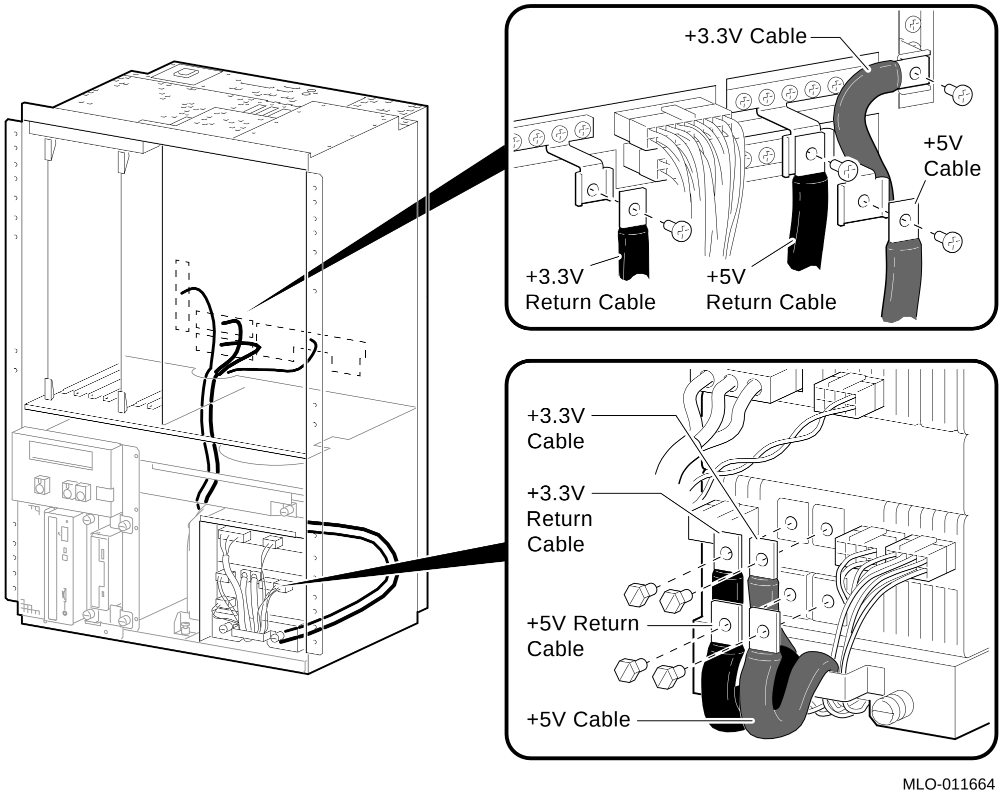


Figure 6–17 Remote I/O Cable (60-pin)

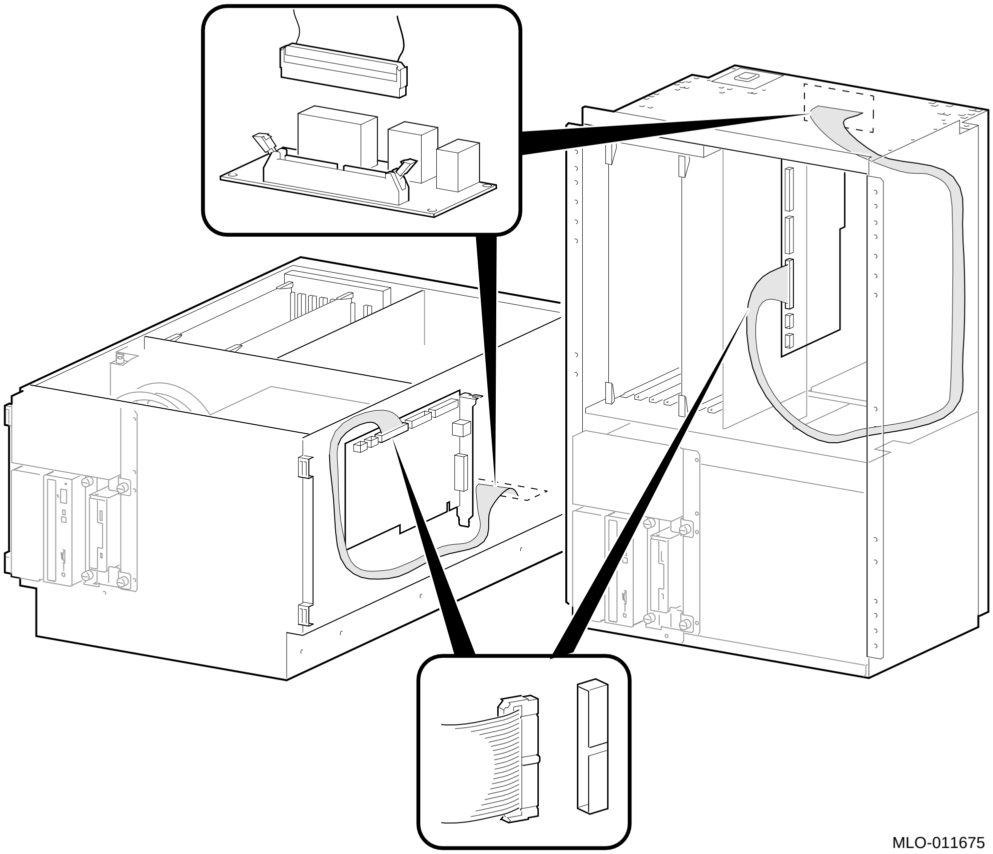


Figure 6–18 SCSI Multinode Cable (50-Pin)

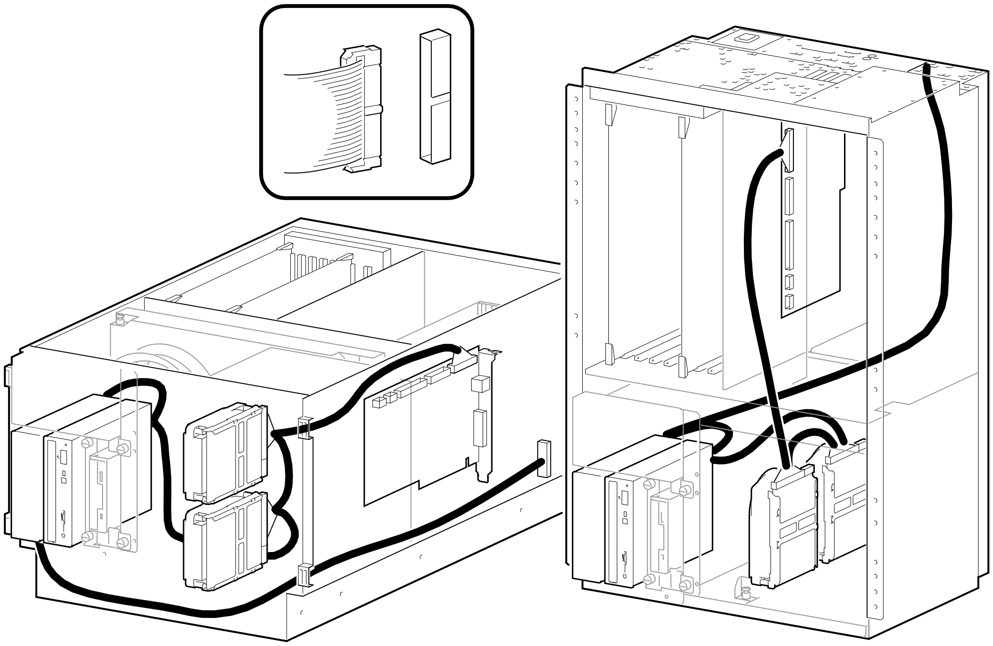
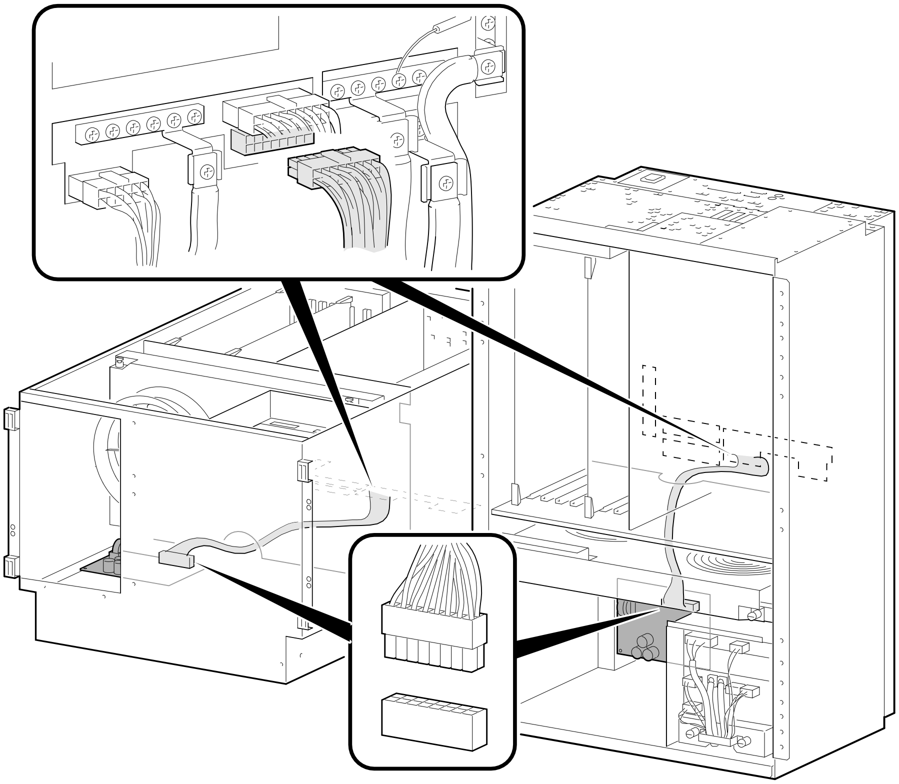


Figure 6–19 -12 V Converter to Backplane Cable



6.2.4 CPU Modules

Note

Different CPU types cannot be used within the same system. Example: A KN450 CPU module and a KN460 CPU module cannot be used in the same system.

Before replacing a CPU module, perform the following steps to verify which CPU is failing. After installing a new CPU, repeat this procedure to ensure that the new CPU configuration is working properly.

STEP 1: CHECK FOR ERRORS LOGGED TO THE CPU.

Verify that errors have been logged through the serial control bus before replacing a CPU module. Using the `show fru` and `show error` console commands, you can determine if errors are logged for a bad CPU.

If an event is logged for any other test than test number 00, the CPU should be replaced.

1. **Enter the** `show fru` **command to check for test-directed diagnostic (TDD) errors logged to the CPU.**

   In the following example, a test-directed diagnostic (TDD) error is logged for CPU0.

```
P00>>> show fru

                                 Rev                     Events logged
 Slot   Option  Part#           Hw Sw   Serial#         SDD     TDD
   0    IO      B2110-AA        C4 0    KA347DWV06      00      00
   2    CPU0    B2020-AA        B2 9    ML33900048      00      01
   3    CPU1    B2020-AA        B2 9    KA34509090      00      00
   6    MEM2    B2022-CA        A1 0    ML34100009      00      00
   7    MEM3    B2022-CA        A1 0    ML34100008      00      00
        .
        .
        .
P00>>>
```

2. **Enter the** `show error cpu0` **command to verify that an error, other than test number 00, is currently logged for that CPU.**

```
P00>>> show error cpu0

CPU0 Module EEROM Event Log

Test Directed Errors
```

```
Entry: 0  Test Number: 02       Subtest Number: 02
Parameter 1: 00000000,00000010
Parameter 2: ffffffff,ffffffff
Parameter 3: fffffeff,ffffffff

CPU Event Counters
CPU Event Counters
C3_CA_NOACK     0
   .
   .
   .
C3_DT_PAR_E     0
C3_DT_PAR_O     0

B-Cache Correctable Errors

Entry   Syndrome        Offset L        Offset H        Count

No Entries Found
P00>>>
```

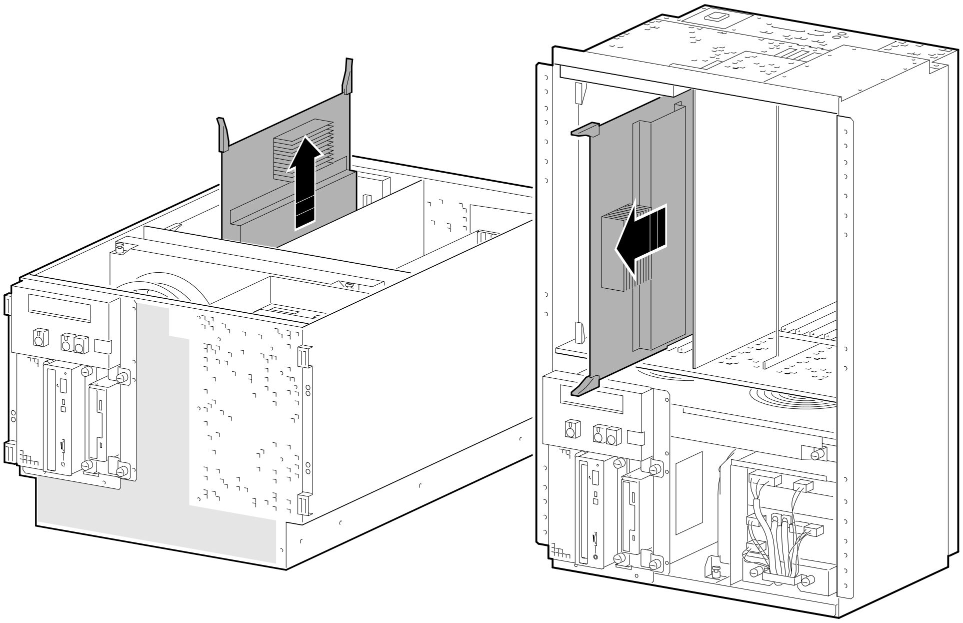
STEP 2: IF THE CPU HAS AN ERROR LOGGED, OTHER THAN FOR TEST NUMBER 00, PERFORM POWER SHUTDOWN AND REPLACE THE CPU MODULE.

An event logged for test number 00 does not indicate a bad CPU. Test number 00 indicates that a CPU failover occurred sometime in the past.

Note

All systems must have a CPU module installed in system bus slot 2 (CPU0).

Figure 6–20 Removing CPU Modules



Warning: CPU and memory modules have parts that operate at high temperatures. Wait 2 minutes after power is removed before handling these modules.

STEP 3: VERIFY THAT ERRORS ARE NO LONGER LOGGED FOR THE CPU.

Use the `show fru` command to verify that the errors are cleared.

```
P00>>> show fru
```

```
                                 Rev                    Events logged
 Slot   Option  Part#           Hw Sw   Serial#         SDD     TDD
   0    IO      B2110-AA        C4 0    KA347DWV06      00      00
   2    CPU0    B2020-AA        B2 9    ML33900048      00      00
   3    CPU1    B2020-AA        B2 9    KA34509090      00      00
   6    MEM2    B2022-CA        A1 0    ML34100009      00      00
   7    MEM3    B2022-CA        A1 0    ML34100008      00      00
        .
        .
        .
        .
        .
        .
P00>>>
```

Note

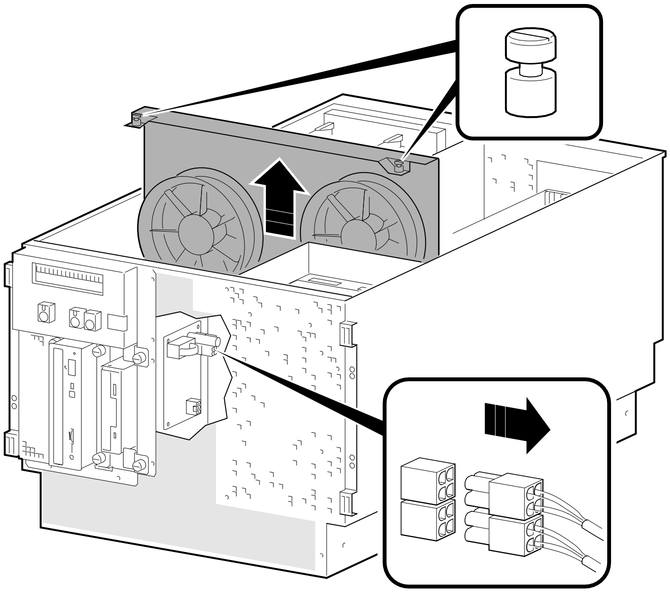
To clear an event logged for test number 00 (CPU failover), use the `clear_error cpu#` command.

6.2.5 Fans (Drawer-Mount)

STEP 1: UNPLUG FANS.

STEP 2: REMOVE FAN ASSEMBLY AND REPLACE FAILING FAN.

Figure 6–21 Removing Fans (Drawer-Mount)

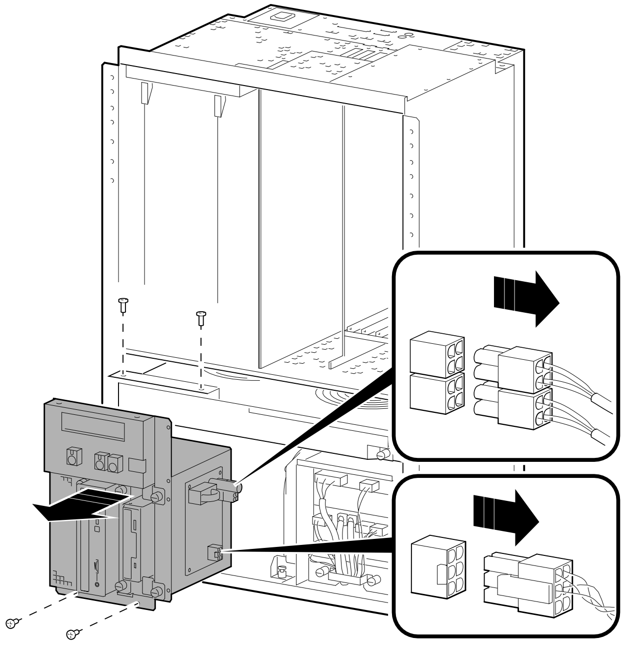


6.2.6 Fans (Vertical-Mount)

STEP 1: UNPLUG FAN CABLES AND STORAGE POWER CABLE.

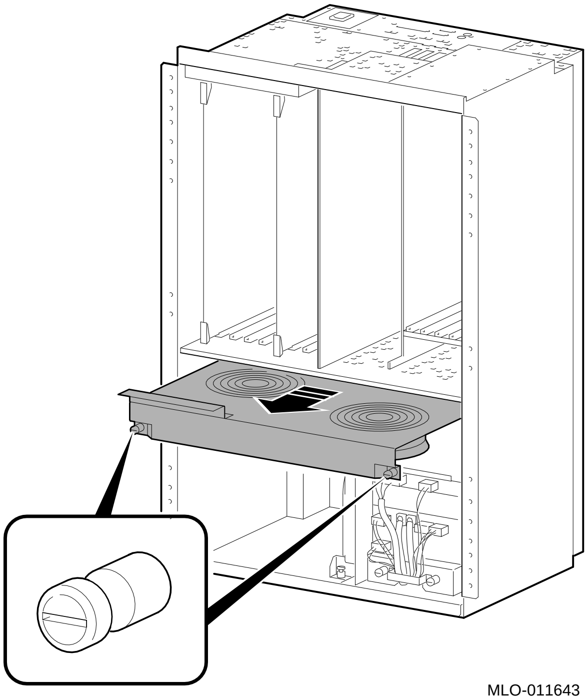
STEP 2: REMOVE OCP AND REMOVABLE STORAGE CHASSIS.

Figure 6–22 Unplugging Cables and Removing OCP Chassis (Vertical-Mount)



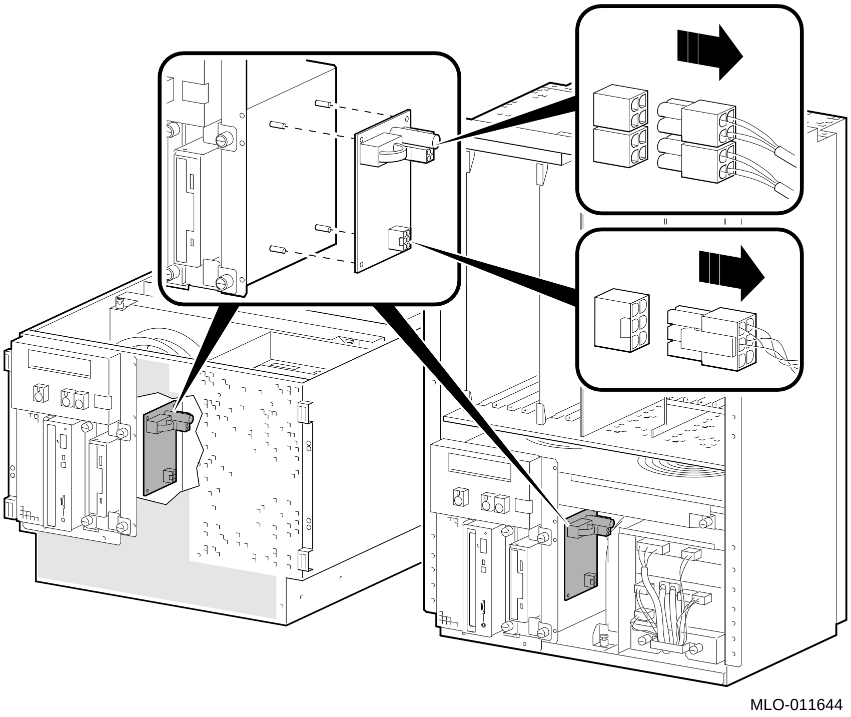
STEP 3: REMOVE FAN ASSEMBLY AND REPLACE FAILING FAN.

Figure 6–23 Removing Fans (Vertical-Mount)



6.2.7 Fan Speed Control Board

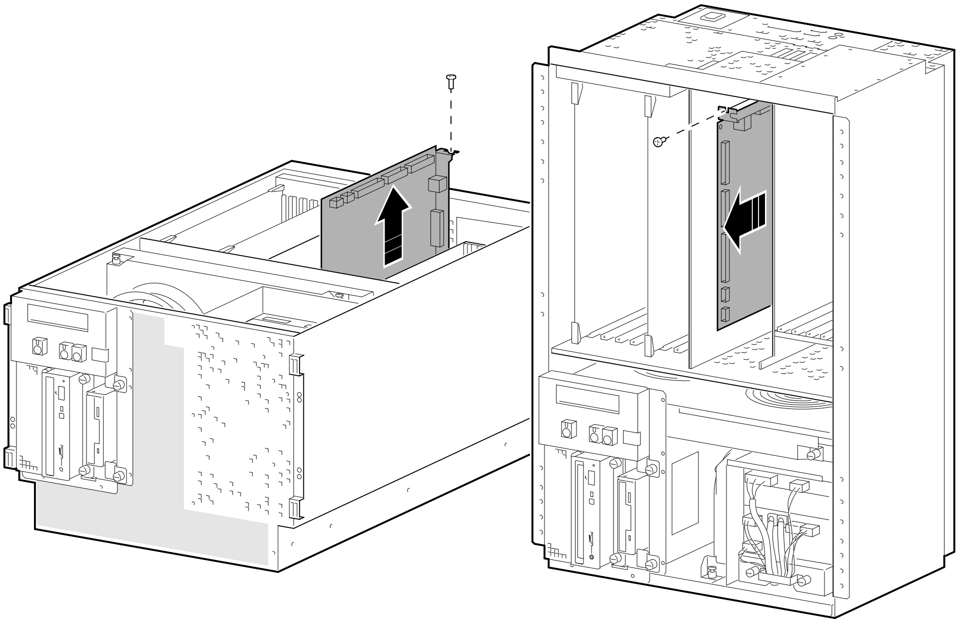
Figure 6–24 Removing Fan Speed Control Board



6.2.8 Standard I/O Module

STEP 1: DISCONNECT THE CABLES AND REMOVE THE MODULE.

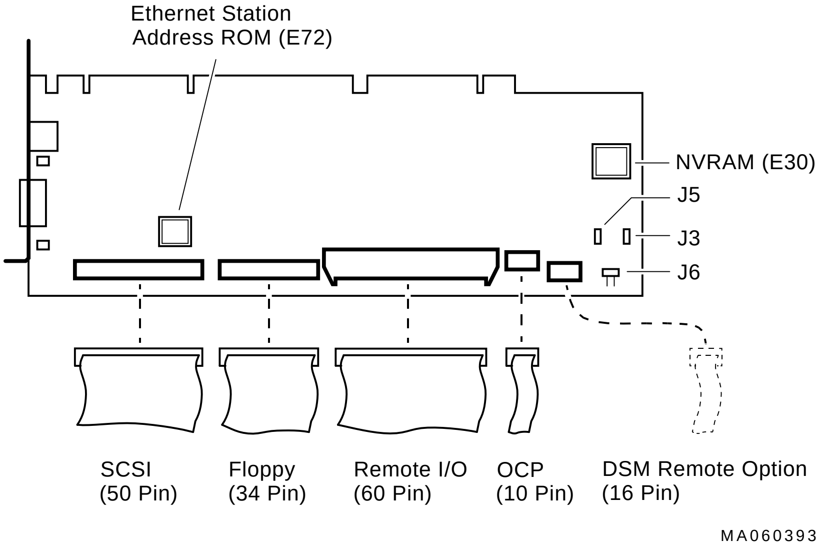
Figure 6–25 Removing Standard I/O Module



STEP 2: MOVE CHIPS TO NEW MODULE.

Move the socketed Ethernet station address ROM (position E72) and NVRAM chip (position E30) to the replacement standard I/O module and set jumpers to match previous settings.

Figure 6–26 Standard I/O Module: Jumpers, Connectors, and Swapable Chips



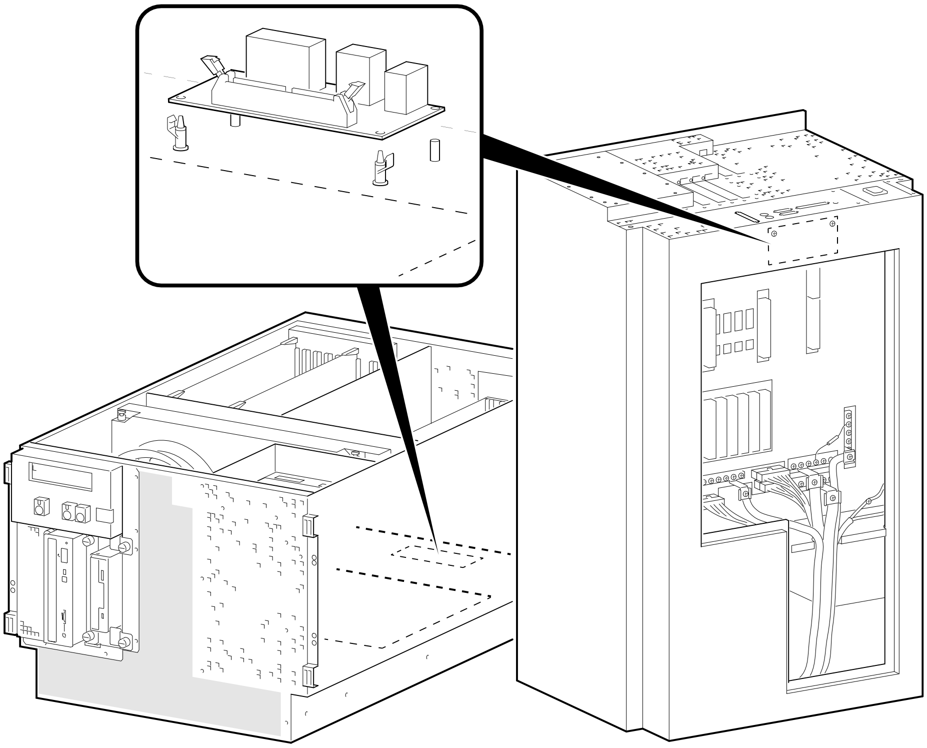
J3–Power supply mode: Digital Alpha VME 2100 systems use the full power mode setting (jumper not installed).
J5–Program voltage: Internal use only.
J6–Fail-Safe: When installed, selects the fail-safe loader firmware.

STEP 3: RUN ECU.

Run the ECU to disable the EISA-based VGA option on the replacement standard I/O module.

6.2.9 Remote I/O Module

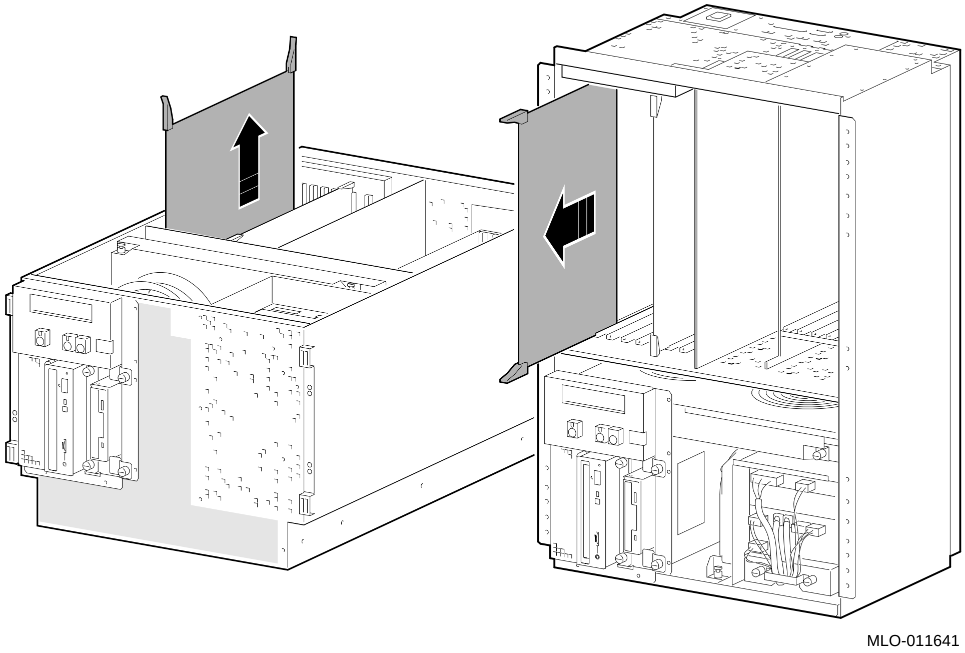
Figure 6–27 Removing Remote I/O Module



6.2.10 Memory Modules

Warning: Memory and CPU modules have parts that operate at high temperatures. Wait 2 minutes after power is removed before handling these modules.

Figure 6–28 Removing Memory Modules

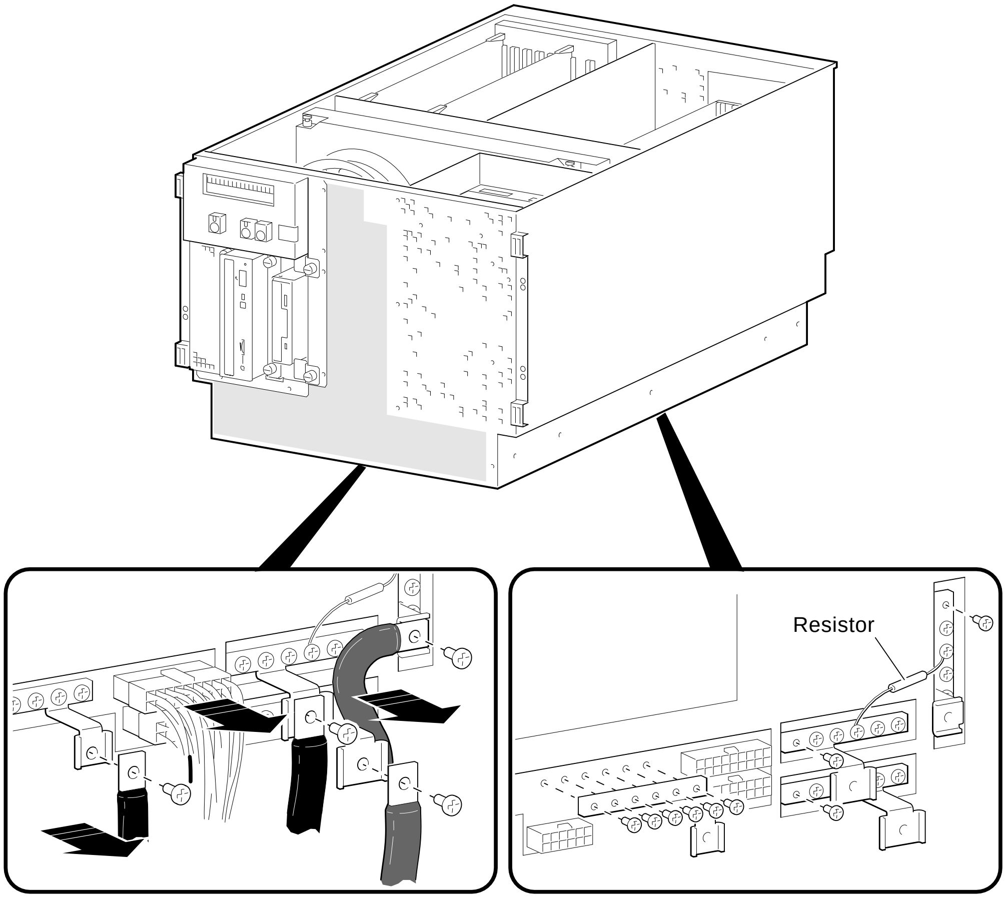


6.2.11 Motherboard (Drawer-Mount)

STEP 1: REMOVE ALL POWER SUPPLY CABLES, RESISTOR, AND POWER BUS BARS FROM MOTHERBOARD.

From the bottom cover, remove the power supply control and 12V cables from their connectors beneath the motherboard. Unscrew the 3V and 5V leads from the power bus bars. Note the position of the resistor between the 3.3 V and logic ground bus bars. Remove the resistor and power bus bars.

Figure 6–29 Removing Power Supply Cables and Power Bus Bars from Motherboard



STEP 2: REMOVE CPU AND MEMORY MODULES.

Warning: Memory and CPU modules have parts that operate at high temperatures. Wait 2 minutes after power is removed before handling these modules.

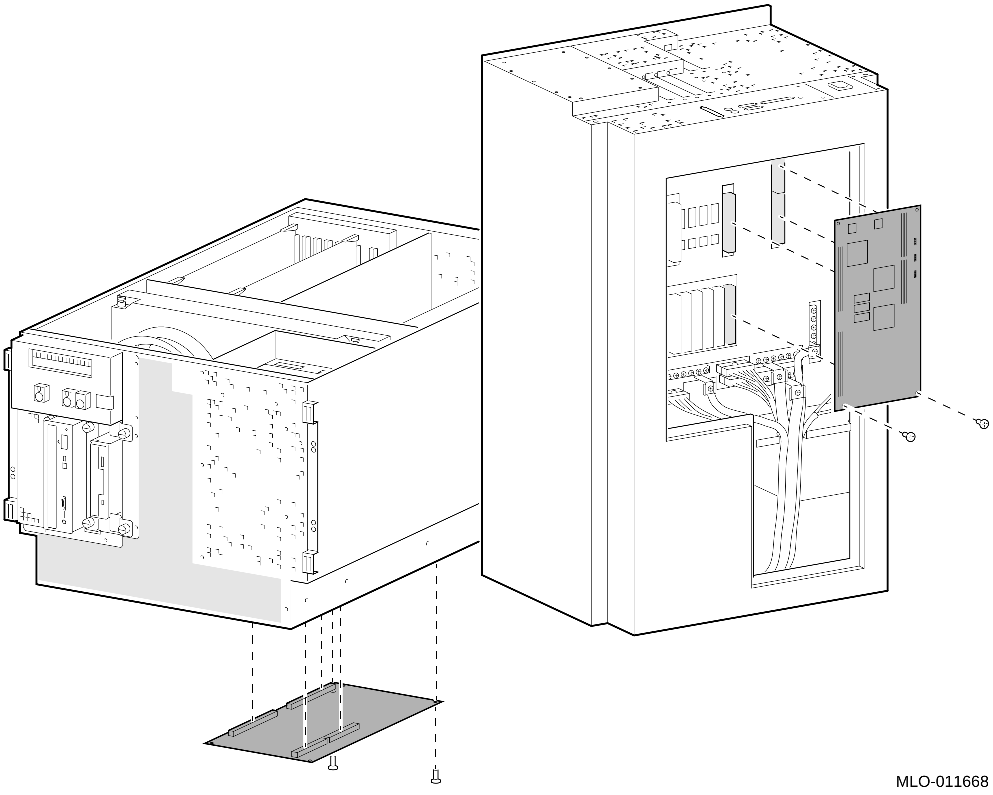
STEP 3: REMOVE ANY CABLES EXITING THE PCI and VME BUS COMPARTMENTS.

Unplug and remove the standard I/O cables from the PCI/VME bus compartment. Remove any other cables that exit from the PCI/VME bus compartment.

STEP 4: REMOVE STANDARD I/O MODULE AND ALL PCI AND VME OPTIONS.

STEP 5: REMOVE THE PCI TO VME DAUGHTER BOARD.

Figure 6–30 Removing PCI to VME Daughter Board



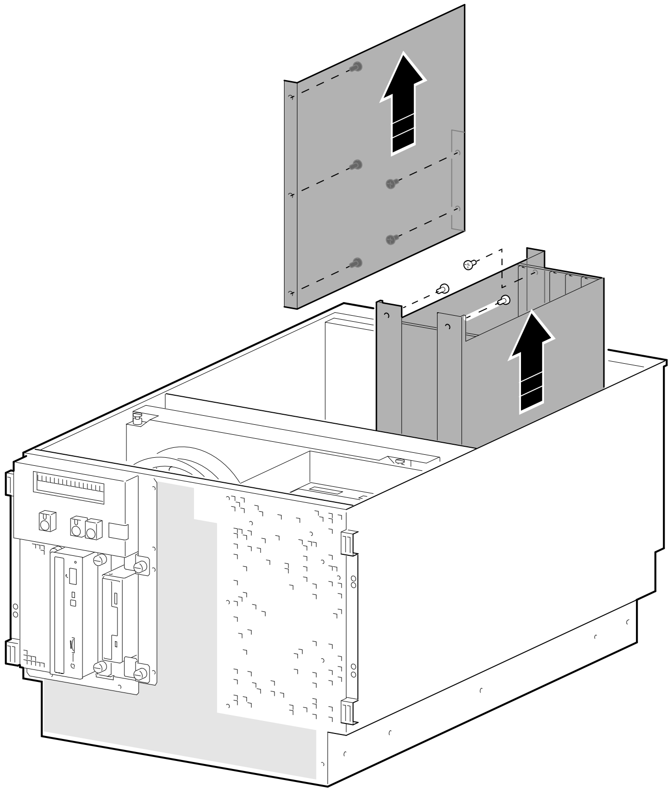
Note

The right screw that secures the PCI to VME daughter board also secures a ground strap (70-32476). This ground strap connects to the second screw from the left on the horizontal power bus bar below the daughter board screw.

Be sure the daughter board is properly seated and the board is contacting its stand offs.

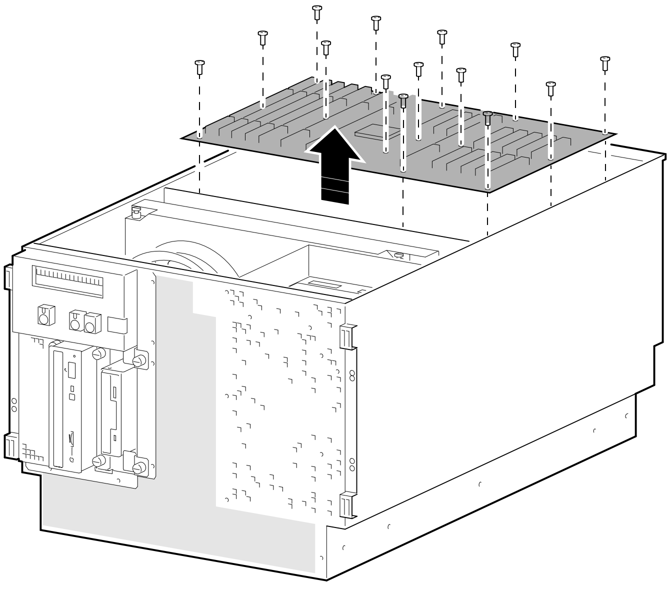
STEP 6: REMOVE THE VME CARD CAGE AND CHASSIS MIDPLATE.

Figure 6–31 Removing VME Card Cage and Chassis Midplate



STEP 7: REMOVE SYSTEM BUS MOTHERBOARD.

Figure 6–32 Removing System Bus Motherboard

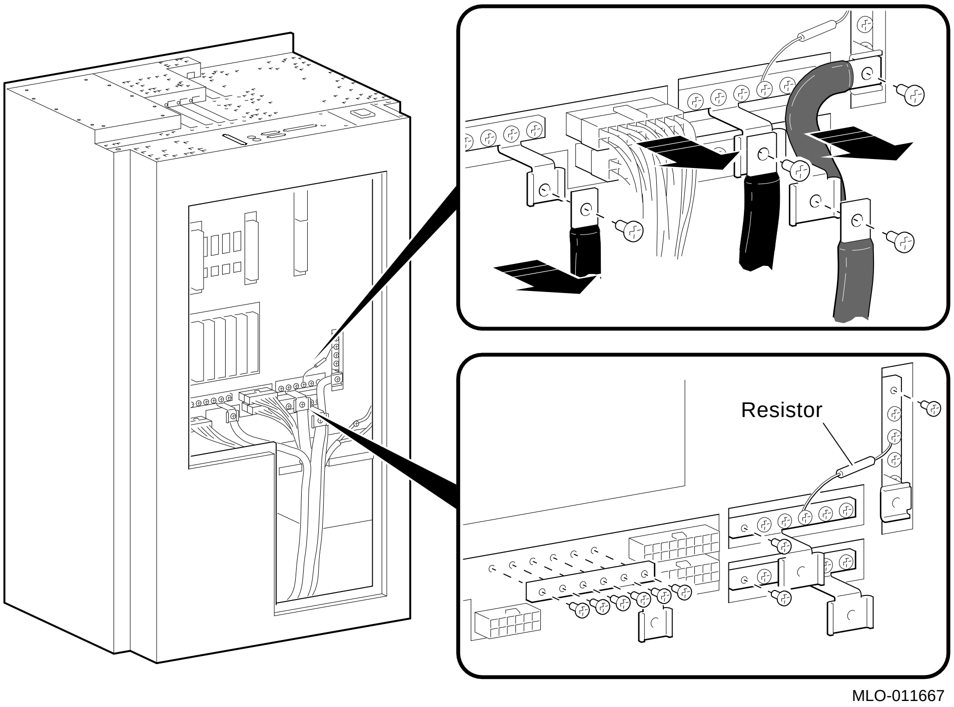


6.2.12 Motherboard (Vertical-Mount)

STEP 1: REMOVE ALL POWER SUPPLY CABLES, RESISTOR, AND POWER BUS BARS FROM MOTHERBOARD.

From the rear cover, remove the power supply control and 12V cables from their connectors beneath the motherboard. Unscrew the 3V and 5V leads from the power bus bars. Note the position of the resistor between the 3.3 V and logic ground bus bars. Remove the resistor and bus bars.

Figure 6–33 Removing Power Supply Cables and Power Bus Bars from Motherboard (Vertical-Mount)



STEP 2: REMOVE CPU AND MEMORY MODULES.

Warning: Memory and CPU modules have parts that operate at high temperatures. Wait 2 minutes after power is removed before handling these modules.

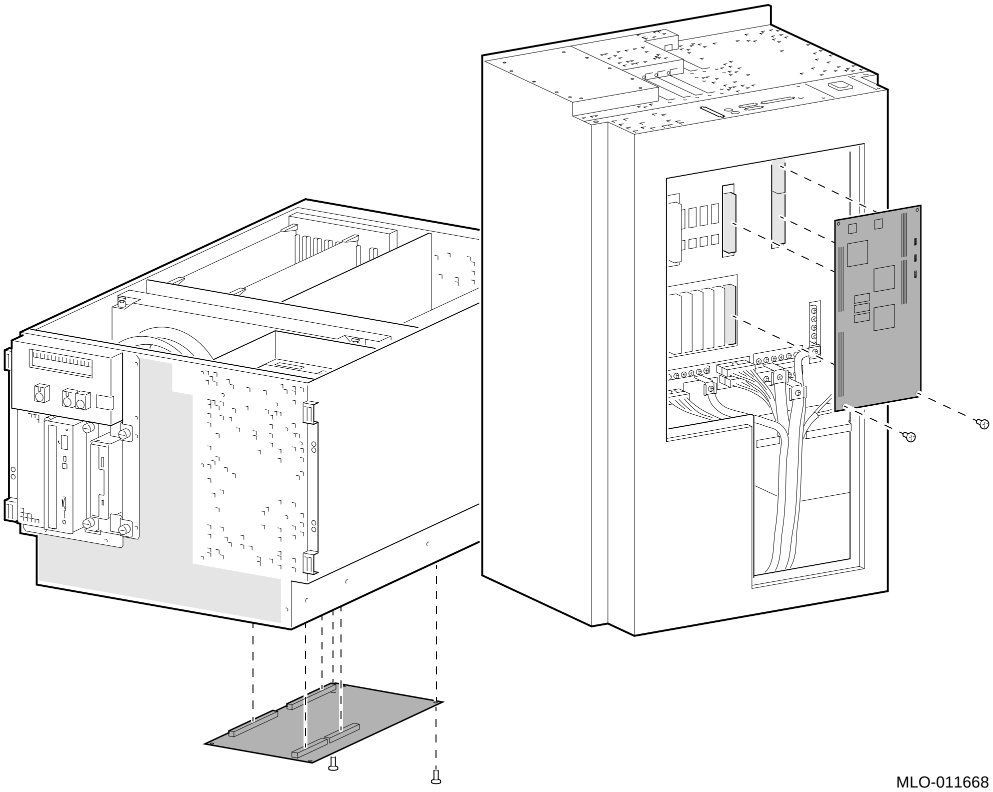
STEP 3: REMOVE ANY CABLES EXITING THE PCI and VME BUS COMPARTMENTS.

Unplug and remove the standard I/O cables from the PCI/VME bus compartment. Remove any other cables that exit from the PCI/VME bus compartment.

STEP 4: REMOVE STANDARD I/O MODULE AND ALL PCI AND VME OPTIONS.

STEP 5: REMOVE THE PCI TO VME DAUGHTER BOARD.

Figure 6–34 Removing PCI to VME Daughter Board



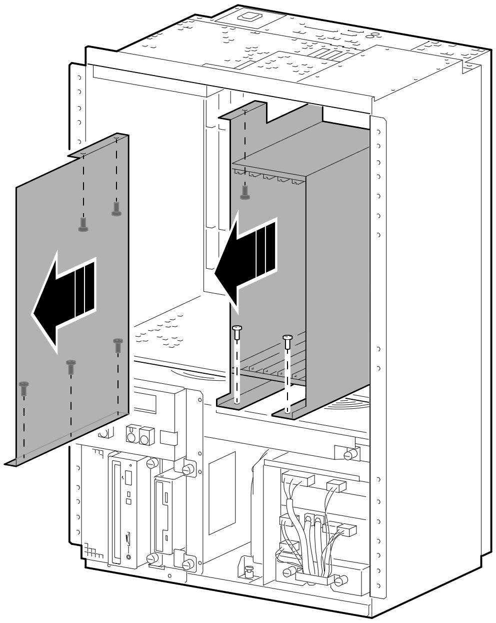
Note

The right screw that secures the PCI to VME daughter board also secures a ground strap (70-32476). This ground strap connects to the second screw from the left on the horizontal power bus bar below the daughter board screw.

Be sure the daughter board is properly seated and the board is contacting its stand offs.

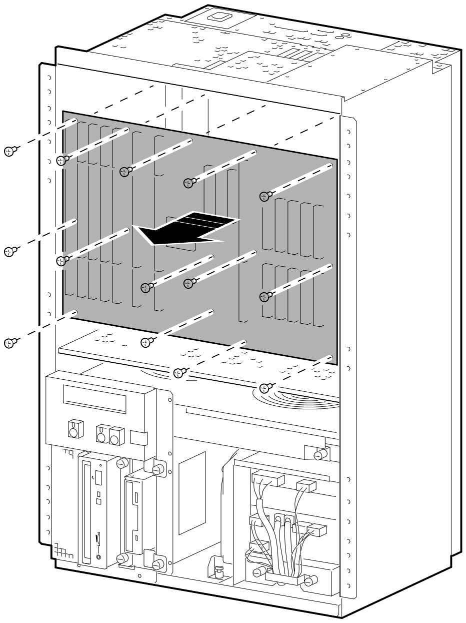
STEP 6: REMOVE THE VME CARD CAGE AND CHASSIS MIDPLATE.

Figure 6–35 Removing VME Card Cage and Chassis Midplate (Vertical-Mount)



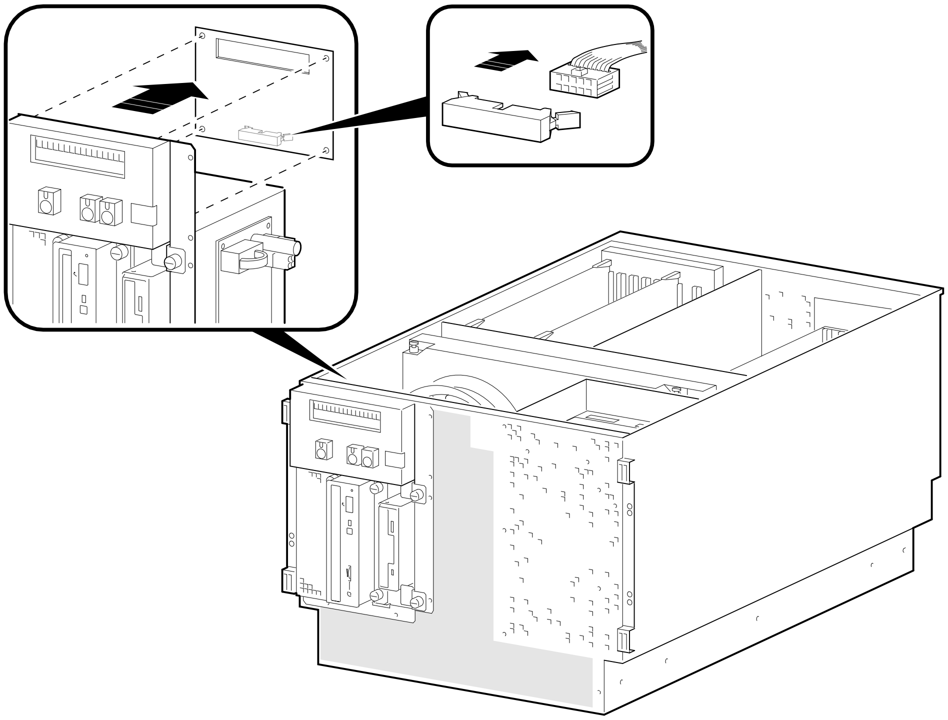
STEP 7: REMOVE SYSTEM BUS MOTHERBOARD.

Figure 6–36 Removing System Bus Motherboard (Vertical-Mount)



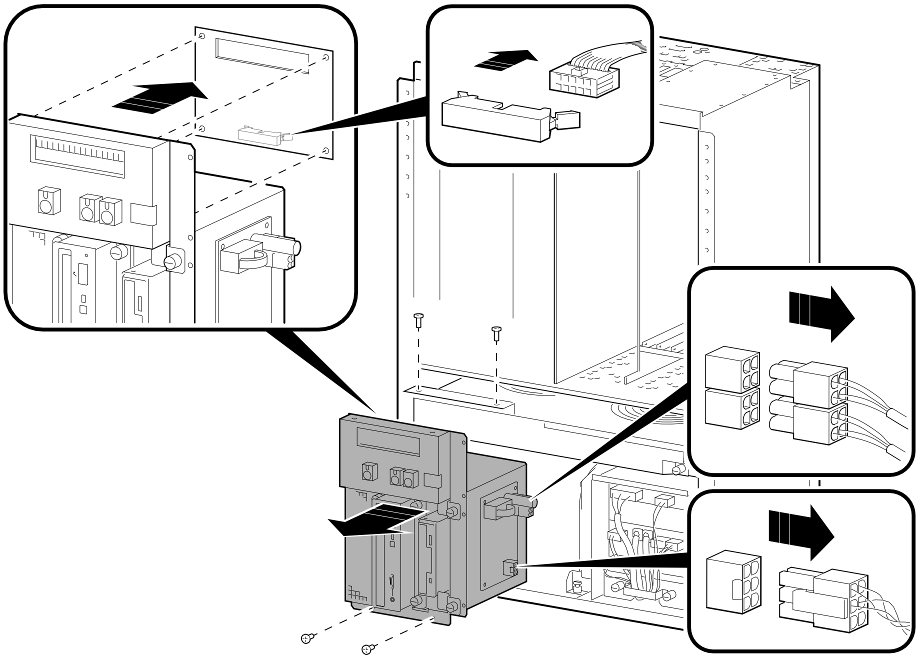
6.2.13 OCP Module (Drawer-Mount)

Figure 6–37 Removing OCP Module (Drawer-Mount)



6.2.14 OCP Module (Vertical-Mount)

Figure 6–38 Removing OCP Module (Drawer-Mount)



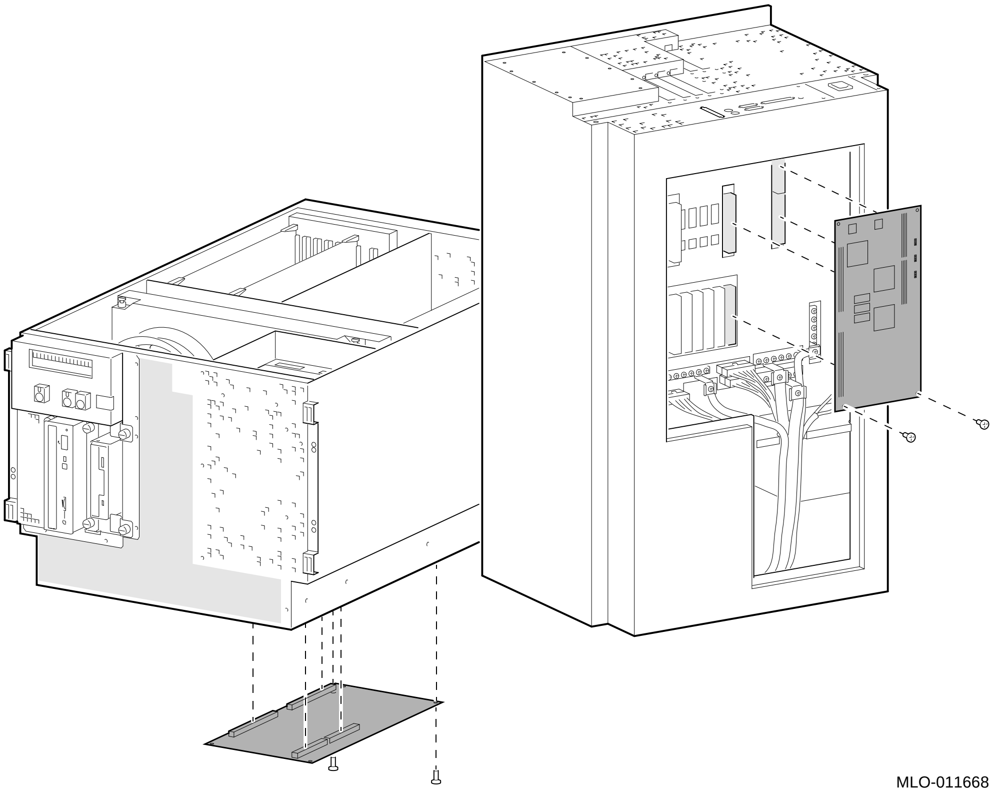
6.2.15 PCI to VME Daughter Board

STEP 1: REMOVE DAUGHTER BOARD.

STEP 2: SET DAUGHTER BOARD JUMPERS TO MATCH PREVIOUS SETTINGS.

Refer to Appendix A for information on the use of the daughter board jumpers.

Figure 6–39 Removing PCI to VME Daughter Board

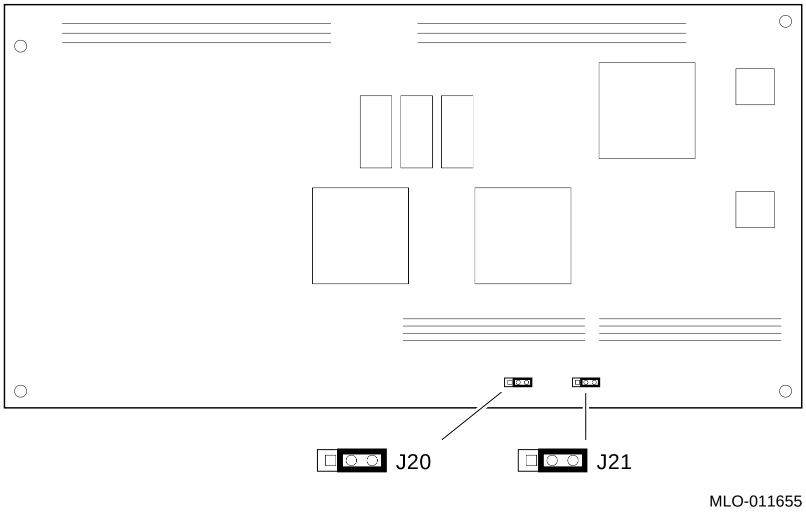


Note

The right screw that secures the PCI to VME daughter board also secures a ground strap (70-32476). This ground strap connects to the second screw from the left on the horizontal power bus bar below the daughter board screw.

Be sure the daughter board is properly seated and the board is contacting its stand offs.

Figure 6–40 PCI to VME Daughter Board Jumpers



Position	J20	J21	Jumper Setting
1	On	On	64 MB (default setting–shown above)
0	Off	Off	
1	Off	On	32 MB
0	On	Off	
1	On	Off	16 MB
0	Off	On	
1	Off	Off	8 MB
0	On	On	

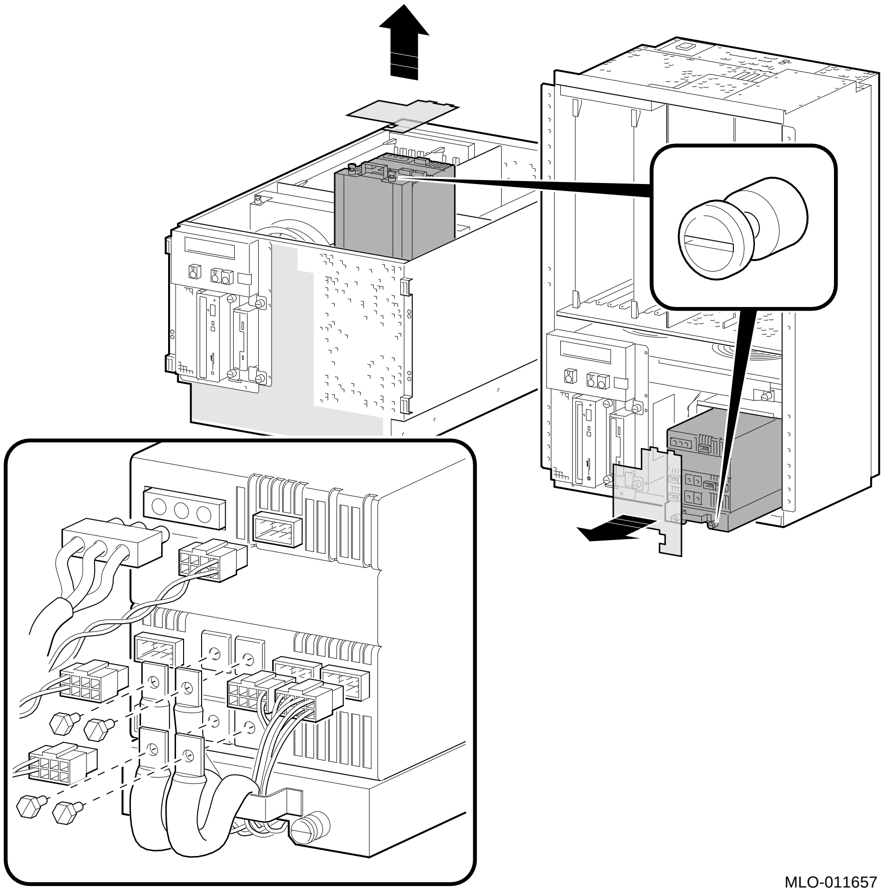
6.2.16 Power Supply

STEP 1: PLASTIC COVER.

STEP 2: REMOVE CABLES.

STEP 3: REMOVE POWER SUPPLY.

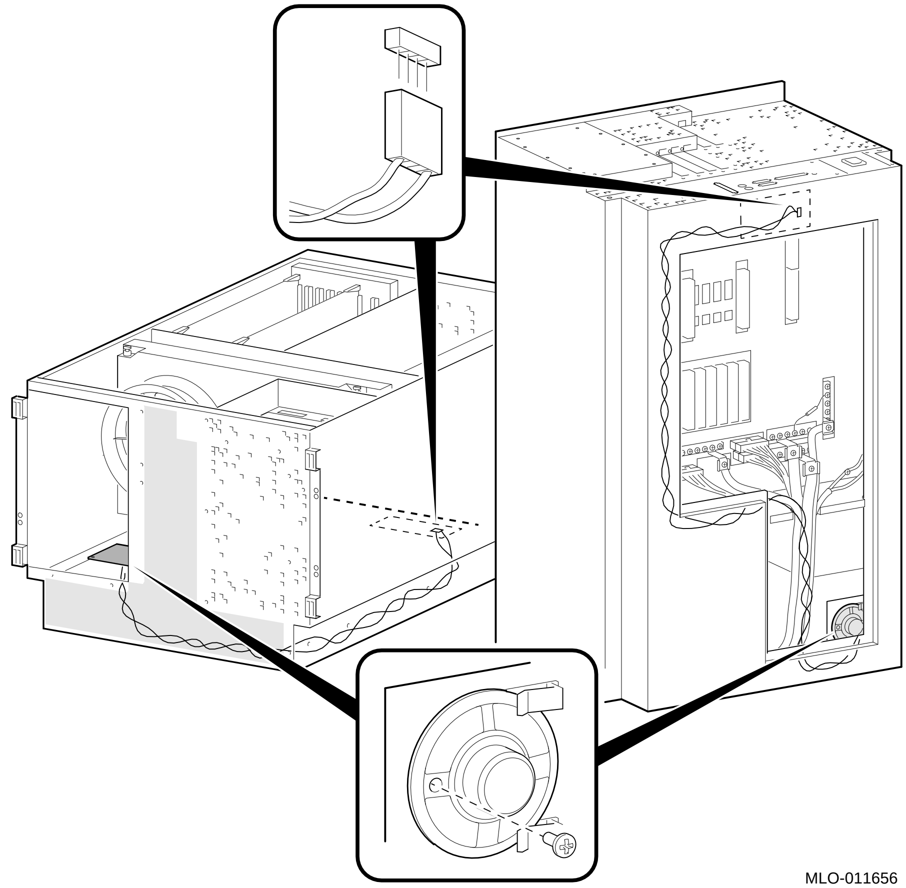
Figure 6–41 Removing Power Supply



ϟ **Warning:** Hazardous voltages contained within. Do not service. Return to factory for service.

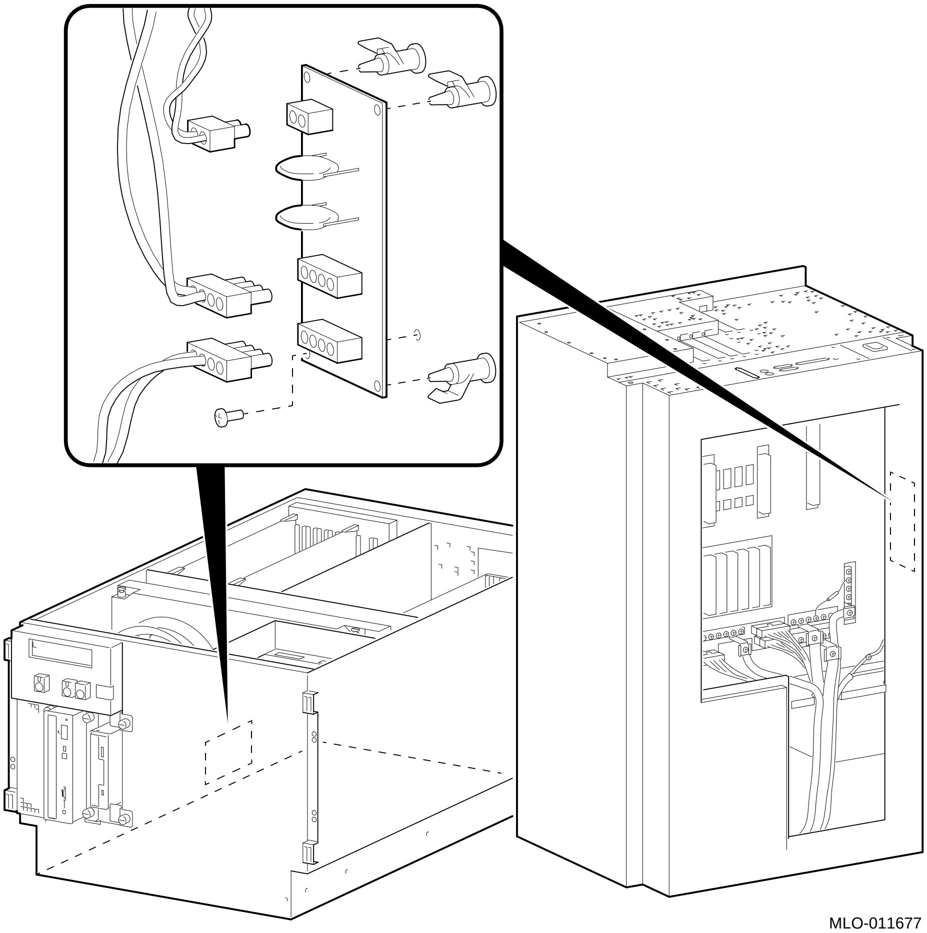
6.2.17 Speaker

Figure 6–42 Removing Speaker



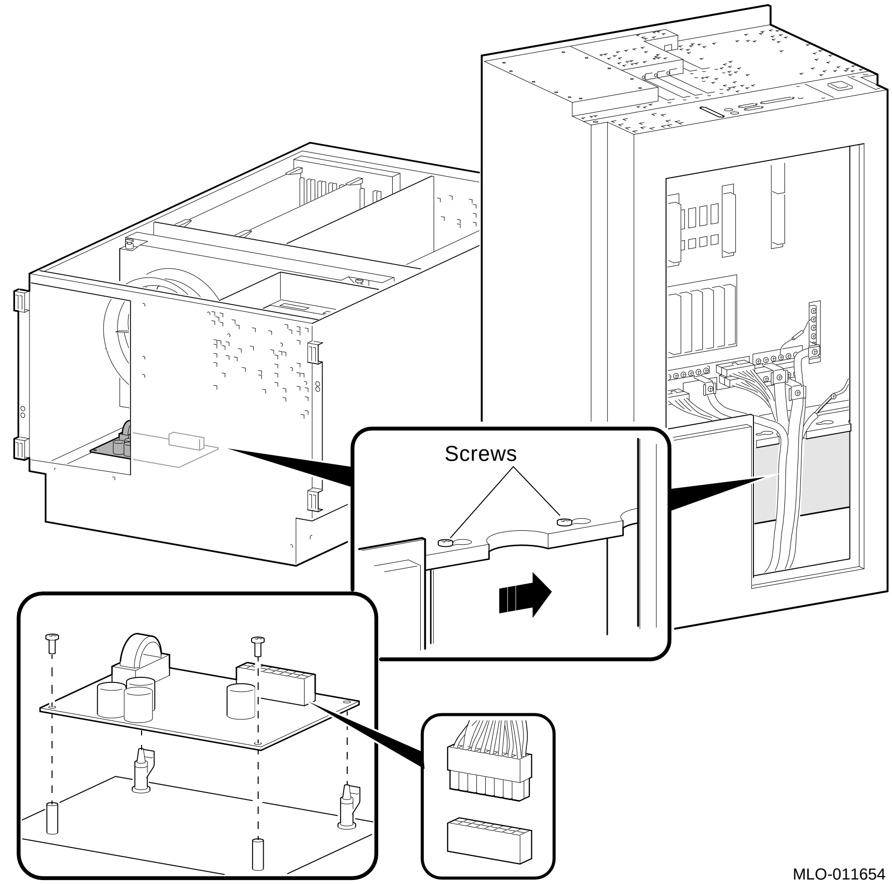
6.2.18 Voltage Protection Module (MOV)

Figure 6–43 Removing Voltage Protection Module



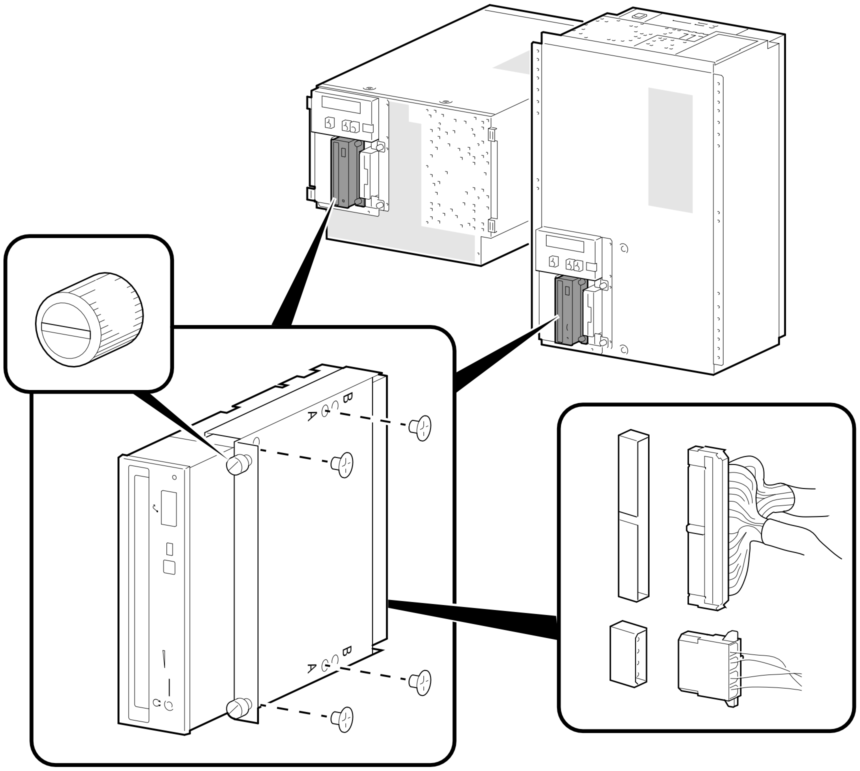
6.2.19 -12 V Converter Module

Figure 6–44 Removing -12 V Converter Module



6.2.20 Removable Media

Figure 6–45 Removing a Removable-Media Drive



MLO-011610a

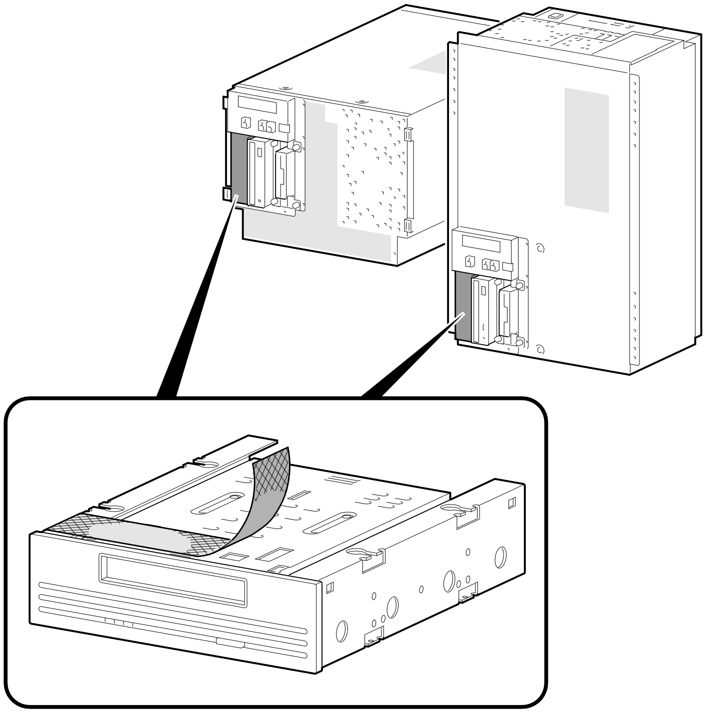
Note

RRDnn and TLZ0n drives use the set of bracket holes marked “A” in Figure 6–45.

The TZK11 drive uses the set of bracket holes marked “B” in Figure 6–45.

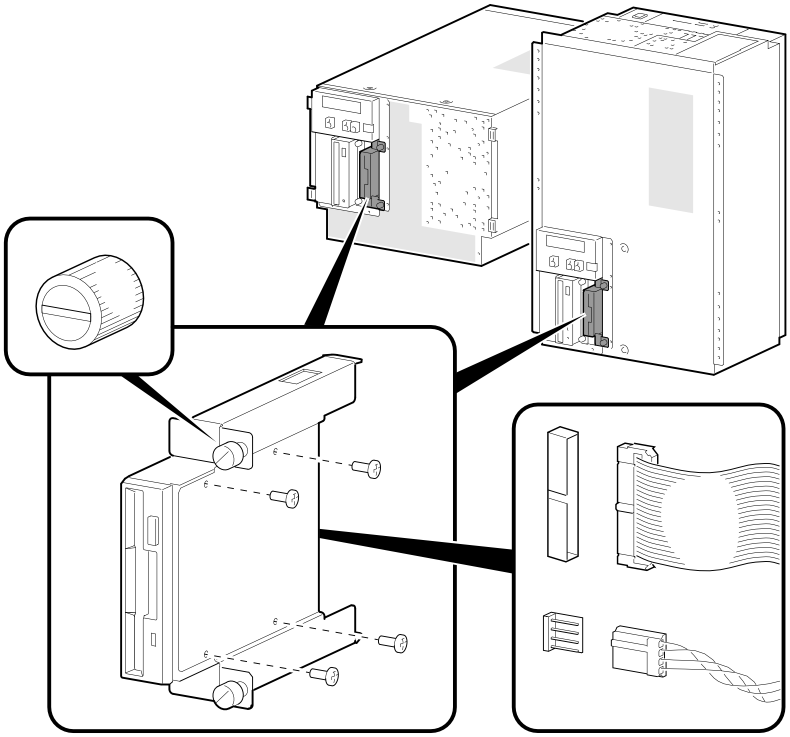
A plastic strip (70-32518-01) must be attached to a TLZ0n tape drive when the tape drive is installed as the left-most removable-media device in a Digital Alpha VME 2100 system with no front bezel (Figure 6–46).

Figure 6–46 Plastic Strip for TLZ0n Tape Drives



MLO-011669

Figure 6–47 Removing Floppy Drive

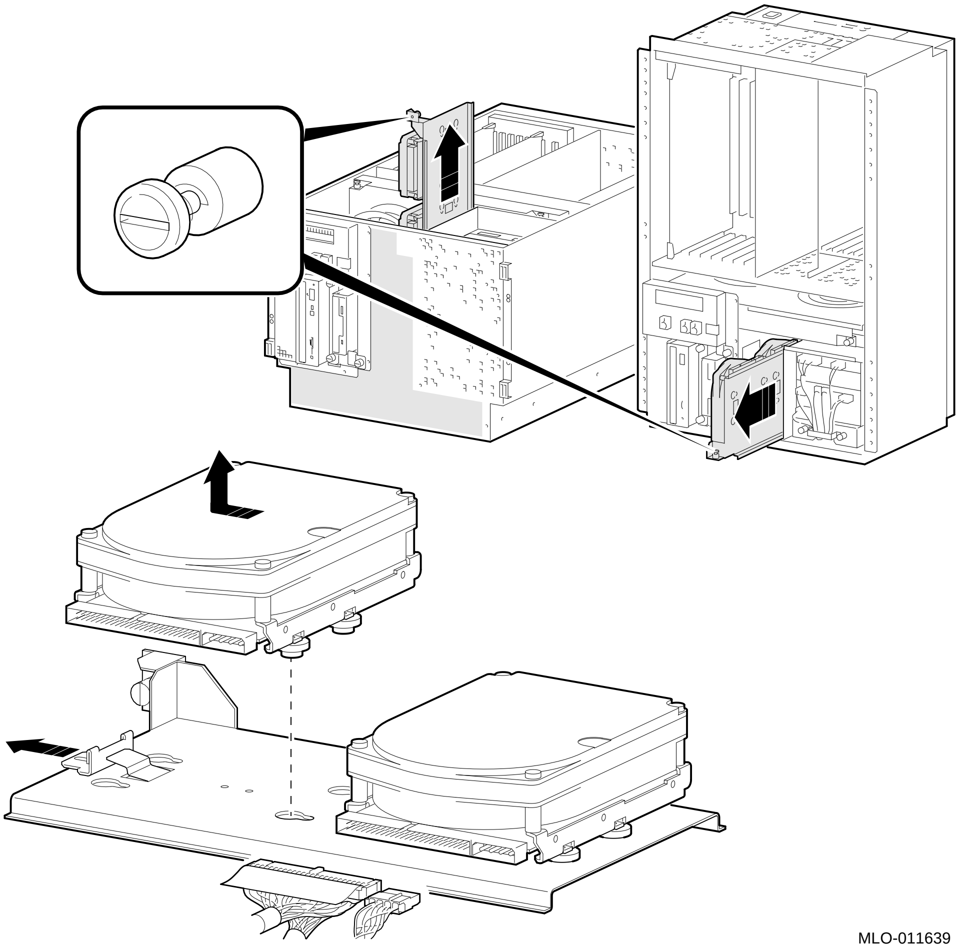


Note

The data cable must be installed opposite to how the connector is keyed, otherwise the drive will not work.

6.2.21 Fixed Disk Drives

Figure 6–48 Removing Fixed Disk Drives



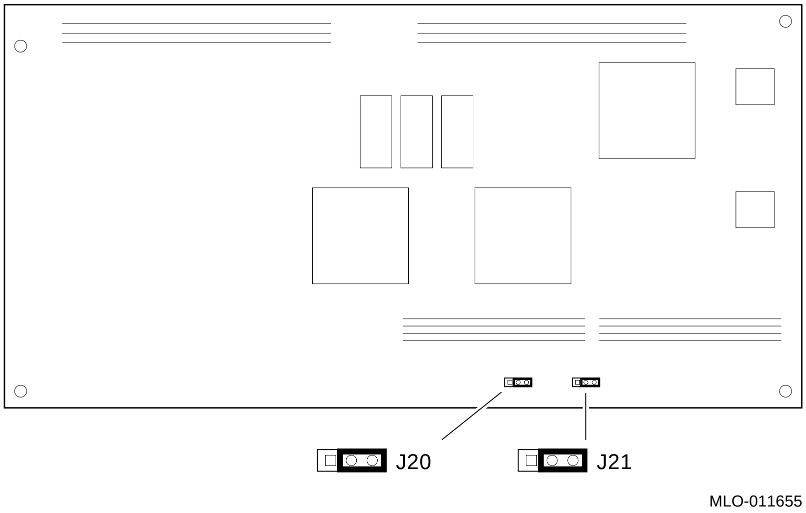
A

VME Daughter Board Jumper Settings

The J20 and J21 jumpers on the VME daughter board are used to set the size of the VME_WINDOW_2 register. The VME_WINDOW_2 register specifies the base of a 64 MB naturally aligned block of PCI memory space used to access the VME bus. This address space is typically re-mapped through scatter gather RAM, onto the VME bus. This address space is aliased to the first 64 MB of the 512 MB VME address space and should be used to allow processor access through "sparse space."

A minimum of 8 MB and a maximum of 64 MB (default setting) can be allocated for the VME_WINDOW_2 register. Use the table following Figure A–1 to determine jumper settings. A jumper must be installed on either position 1 or position 0 of each jumper.

Figure A–1 PCI to VME Daughter Board Jumpers



Position	J20	J21	Jumper Setting
1	On	On	64 MB (default setting—shown above)
0	Off	Off	
1	Off	On	32 MB
0	On	Off	
1	On	Off	16 MB
0	Off	On	
1	Off	Off	8 MB
0	On	On	

Glossary

10BASE-T Ethernet network

IEEE standard 802.3-compliant Ethernet products used for local distribution of data. These networking products characteristically use twisted-pair cable.

ARC

User interface to the console firmware for operating systems that require firmware compliance with the *Windows NT Portable Boot Loader Specification*. ARC stands for Advanced RISC Computing.

AUI Ethernet network

Attachment unit interface. An IEEE standard 802.3-compliant Ethernet network connected with standard Ethernet cable.

autoboot

A system boot initiated automatically by software when the system is powered up or reset.

availability

The amount of scheduled time that a computing system provides application service during the year. Availability is typically measured as either a percentage of uptime per year or as system unavailability, the number of hours or minutes of downtime per year.

BA350 storage shelf

A StorageWorks modular storage shelf used for disk storage in some AlphaServer systems.

BA720 enclosure

The enclosure that houses the AlphaServer 2000 deskside pedestal system.

BA740 enclosure

The enclosure that houses the AlphaServer 2100 large pedestal system.

BA741 enclosure

The enclosure that houses the AlphaServer 2100R rack-mountable system and AlphaServer 2200 cabinet system.

BA742 enclosure

The enclosure that houses the Digital Alpha VME 2100 system.

backplane

The main board or panel that connects all of the modules in a computer system.

backup cache

A second, very fast cache memory that is closely coupled with the processor.

bandwidth

Term used to express the rate of data transfer in a bus or I/O channel. It is expressed as the amount of data that can be transferred in a given time, for example megabytes per second.

battery backup unit

A battery unit that provides power to the entire system enclosure (or to an expander enclosure) in the event of a power failure. Another term for uninterruptible power supply (UPS).

boot

Short for bootstrap. To load an operating system into memory.

boot device

The device from which the system bootstrap software is acquired.

boot flags

A flag is a system parameter set by the user. Boot flags contain information that is read and used by the bootstrap software during a system bootstrap procedure.

boot server

A computer system that provides boot services to remote devices such as network routers.

bootstrap

The process of loading an operating system into memory.

bugcheck

A software condition, usually the response to software's detection of an "internal inconsistency," which results in the execution of the system bugcheck code.

bus

A collection of many transmission lines or wires. The bus interconnects computer system components, providing a communications path for addresses, data, and control information or external terminals and systems in a communications network.

bystander

A system bus node (CPU, standard I/O, or memory) that is not addressed by a current system bus commander.

byte

A group of eight contiguous bits starting on an addressable byte boundary. The bits are numbered right to left, 0 through 7.

cache memory

A small, high-speed memory placed between slower main memory and the processor. A cache increases effective memory transfer rates and processor speed. It contains copies of data recently used by the processor and fetches several bytes of data from memory in anticipation that the processor will access the next sequential series of bytes.

card cage

A mechanical assembly in the shape of a frame that holds modules against the system and storage backplanes.

carrier

The individual container for all StorageWorks devices, power supplies, and so forth. In some cases because of small form factors, more than one device can be mounted in a carrier. Carriers can be inserted in modular shelves. Modular shelves can be mounted in modular enclosures.

CD–ROM

A read-only compact disc. The optical removable media used in a compact disc reader.

central processing unit (CPU)

The unit of the computer that is responsible for interpreting and executing instructions.

client-server computing

An approach to computing whereby a computer—the “server”—provides a set of services across a network to a group of computers requesting those services—the “clients.”

cluster

A group of networked computers that communicate over a common interface. The systems in the cluster share resources, and software programs work in close cooperation.

cold bootstrap

A bootstrap operation following a power-up or system initialization (restart). On Alpha AXP based systems, the console loads PALcode, sizes memory, and initializes environment variables.

commander

In a particular bus transaction, a CPU or standard I/O that initiates the transaction.

command line interface

One of two modes of operation in the AlphaServer operator interface. The command line interface supports the OpenVMS and Digital UNIX operating systems. It allows you to configure and test the system, examine and alter system state, and boot the operating system.

console mode

The state in which the system and the console terminal operate under the control of the console program.

console program

The code that the executes during console mode.

console subsystem

The subsystem that provides the user interface for a computer system when the operating system is not running.

console terminal

The terminal connected to the console subsystem. It is used to start the system and direct activities between the computer operator and the console subsystem.

CPU failover

On multiprocessor systems, functionality that allows the system to power up and boot the operating system even if only one CPU is working.

data bus

A bus used to carry data between two or more components of the system.

data cache

A high-speed cache memory reserved for the storage of data. Abbreviated as D-cache.

DECchip 21064 processor

The CMOS, single-chip processor based on the Alpha AXP architecture and used on many AlphaGeneration computers.

Digital UNIX for AXP systems

Digital UNIX is an X/Open UNIX 93 branded product. Digital UNIX runs on the range of AlphaGeneration systems, from workstations to servers.

DEC VET

Digital DEC Verifier and Exerciser Tool. A multipurpose system diagnostic tool that performs exerciser-oriented maintenance testing.

diagnostic program

A program that is used to find and correct problems with a computer system.

direct-mapping cache

A cache organization in which only one address comparison is needed to locate any data in the cache, because any block of main memory data can be placed in only one possible position in the cache.

direct memory access (DMA)

Access to memory by an I/O device that does not require processor intervention.

DRAM

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

DSSI

Digital's proprietary data bus that uses the System Communication Architecture (SCA) protocols for direct host-to-storage communications.

DSSI cluster

A cluster system that uses the DSSI bus as the interconnect between DSSI disks and systems.

DUP server

Diagnostic Utility Program server. A firmware program on-board DSSI devices that allows a user to set host to a specified device in order to run internal tests or modify device parameters.

ECC

Error correction code. Code and algorithms used by logic to facilitate error detection and correction.

EEPROM

Electrically erasable programmable read-only memory. A memory device that can be byte-erased, written to, and read from.

EISA bus

Extended Industry Standard Architecture bus. A 32-bit industry-standard I/O bus used primarily in high-end PCs and servers.

EISA Configuration Utility (ECU)

A feature of the EISA bus that helps you select a conflict-free system configuration and perform other system services. The ECU must be run whenever you change, add, or remove an EISA or ISA controller.

environment variables

Global data structures that can be accessed only from console mode. The setting of these data structures determines how a system powers up, boots the operating system, and operates.

Ethernet

IEEE 802.3 standard local area network.

ERF/UERF

Error Report Formatter. ERF is used to present error log information for OpenVMS. UERF is used to present error log information for Digital UNIX.

Factory Installed Software (FIS)

Operating system software that is loaded into a system disk during manufacture. On site, the FIS is bootstrapped in the system.

fail-safe loader (FSL)

A program that allows you to power up without initiating drivers or running power-up diagnostics. From the fail-safe loader you can perform limited console functions.

Fast SCSI

An optional mode of SCSI-2 that allows transmission rates of up to 10 megabytes per second.

FDDI

Fiber Distributed Data Interface. A high-speed networking technology that uses fiber optics as the transmissions medium.

FIB

Flexible interconnect bridge. A converter that allows the expansion of the system enclosure to other DSSI devices and systems.

field-replaceable unit

Any system component that a qualified service person is able to replace on site.

firmware

Software code stored in hardware.

fixed-media compartments

Compartments that house nonremovable storage media.

Flash ROM

Flash-erasable programmable read-only memory. Flash ROMs can be bank- or bulk-erased.

FRU

Field-replaceable unit. Any system component that a qualified service person is able to replace on site.

full-height device

Standard form factor for 5 1/4-inch storage devices.

half-height device

Standard form factor for storage devices that are not the height of full-height devices.

halt

The action of transferring control of the computer system to the console program.

hose

The interface between the card cage and the I/O subsystems.

hot swap

The process of removing a device from the system without shutting down the operating system or powering down the hardware.

initialization

The sequence of steps that prepare the computer system to start. Occurs after a system has been powered up.

instruction cache

A high-speed cache memory reserved for the storage of instructions. Abbreviated as I-cache.

interrupt request lines (IRQs)

Bus signals that connect an EISA or ISA module (for example, a disk controller) to the system so that the module can get the system's attention via an interrupt.

I/O backplane

One of two backplanes on the AlphaServer 2000 system. The I/O backplane contains three PCI option slots and seven EISA option slots. It also contains a SCSI channel, diskette controller, two serial ports, and a parallel printer port.

ISA

Industry Standard Architecture. An 8-bit or 16-bit industry-standard I/O bus, widely used in personal computer products. The EISA bus is a superset of the ISA bus.

LAN

Local area network. A high-speed network that supports computers that are connected over limited distances.

latency

The amount of time it takes the system to respond to an event.

LED

Light-emitting diode. A semiconductor device that glows when supplied with voltage. A LED is used as an indicator light.

loopback test

Internal and external tests that are used to isolate a failure by testing segments of a particular control or data path. A subset of ROM-based diagnostics.

machine check/interrupts

An operating system action triggered by certain system hardware-detected errors that can be fatal to system operation. Once triggered, machine check handler software analyzes the error.

mass storage device

An input/output device on which data is stored. Typical mass storage devices include disks, magnetic tapes, and CD–ROM.

MAU

Medium attachment unit. On an Ethernet LAN, a device that converts the encoded data signals from various cabling media (for example, fiber optic, coaxial, or ThinWire) to permit connection to a networking station.

memory interleaving

The process of assigning consecutive physical memory addresses across multiple memory controllers. Improves total memory bandwidth by overlapping system bus command execution across multiple memory modules.

menu interface

One of two modes of operation in the AlphaServer operator interface. Menu mode lets you boot and configure the Windows NT operating system by selecting choices from a simple menu. The EISA Configuration Utility is also run from the menu interface.

modular shelves

In the StorageWorks modular subsystem, a shelf contains one or more modular carriers, generally up to a limit of seven. Modular shelves can be mounted in system enclosures, in I/O expansion enclosures, and in various StorageWorks modular enclosures.

MOP

Maintenance Operations Protocol. A transport protocol for network bootstraps and other network operations.

motherboard

The main circuit board of a computer. The motherboard contains the base electronics for the system (for example, base I/O, CPU, ROM, and console serial line unit) and has connectors where options (such as I/Os and memories) can be plugged in.

multiprocessing system

A system that executes multiple tasks simultaneously.

node

A device that has an address on, is connected to, and is able to communicate with other devices on a bus. Also, an individual computer system connected to the network that can communicate with other systems on the network.

NVRAM

Nonvolatile random-access memory. Memory that retains its information in the absence of power.

OCP

Operator control panel.

open system

A system that implements sufficient open specifications for interfaces, services, and supporting formats to enable applications software to:

- Be ported across a wide range of systems with minimal changes
- Interoperate with other applications on local and remote systems
- Interact with users in a style that facilitates user portability

OpenVMS AXP operating system

A general-purpose multiuser operating system that supports AlphaGeneration computers in both production and development environments. OpenVMS AXP software supports industry standards, facilitating application portability and interoperability. OpenVMS AXP provides symmetric multiprocessing (SMP) support for AXP multiprocessing systems.

operating system mode

The state in which the system console terminal is under the control of the operating system. Also called program mode.

operator control panel

The panel located behind the front door of the system, which contains the power-up/diagnostic display, DC On/Off button, Halt button, and Reset button.

PALcode

Alpha AXP Privileged Architecture Library code, written to support Alpha AXP processors. PALcode implements architecturally defined behavior.

PCI

Peripheral Component Interconnect. An industry-standard expansion I/O bus that is the preferred bus for high-performance I/O options. Available in a 32-bit and a 64-bit version.

portability

The degree to which a software application can be easily moved from one computing environment to another.

porting

Adapting a given body of code so that it will provide equivalent functions in a computing environment that differs from the original implementation environment.

power-down

The sequence of steps that stops the flow of electricity to a system or its components.

power-up

The sequence of events that starts the flow of electrical current to a system or its components.

primary cache

The cache memory that is the fastest and closest to the processor.

processor module

Module that contains the CPU chip.

program mode

The state in which the system console terminal is under the control of a program other than the console program.

RAID

Redundant array of inexpensive disks. A technique that organizes disk data to improve performance and reliability. RAID has three attributes:

- It is a set of physical disks viewed by the user as a single logical device.
- The user's data is distributed across the physical set of drives in a defined manner.
- Redundant disk capacity is added so that the user's data can be recovered even if a drive fails.

redundant

Describes duplicate or extra computing components that protect a computing system from failure.

reliability

The probability a device or system will not fail to perform its intended functions during a specified time.

responder

In any particular bus transaction, memory, CPU, or I/O that accepts or supplies data in response to a command/address from the system bus commander.

RISC

Reduced instruction set computer. A processor with an instruction set that is reduced in complexity.

ROM-based diagnostics

Diagnostic programs resident in read-only memory.

script

A data structure that defines a group of commands to be executed. Similar to a VMS command file.

SCSI

Small Computer System Interface. An ANSI-standard interface for connecting disks and other peripheral devices to computer systems. Some devices are supported under the SCSI-1 specification; others are supported under the SCSI-2 specification.

self-test

A test that is invoked automatically when the system powers up.

serial control bus

A two-conductor serial interconnect that is independent of the system bus. This bus links the processor modules, the I/O, the memory, the power subsystem, and the operator control panel.

serial ROM

In the context of the CPU module, ROM read by the DECchip microprocessor after reset that contains low-level diagnostic and initialization routines.

SIMM

Single in-line memory module.

SMP

Symmetric multiprocessing. A processing configuration in which multiple processors in a system operate as equals, dividing and sharing the workload.

SRM

User interface to console firmware for operating systems that expect firmware compliance with the *Alpha System Reference Manual (SRM).*

standard I/O module

Module that provides a standard set of I/O functions on some AXP servers. It resides in a dedicated slot in the EISA bus card cage.

storage array

A group of mass storage devices, frequently configured as one logical disk.

StorageWorks

Digital's modular storage subsystem (MSS), which is the core technology of the Alpha AXP SCSI-2 mass storage solution. Consists of a family of low-cost mass storage products that can be configured to meet current and future storage needs.

superpipelined

Describes a pipelined processor that has a larger number of pipe stages and more complex scheduling and control.

superscalar

Describes a processor that issues multiple independent instructions per clock cycle.

symmetric multiprocessing (SMP)

A processing configuration in which multiple processors in a system operate as equals, dividing and sharing the workload.

symptom-directed diagnostics (SDDs)

An approach to diagnosing computer system problems whereby error data logged by the operating system is analyzed to capture information about the problem.

system backplane

One of two backplanes on the AlphaServer 2000 system. The system backplane supports up to two CPU modules, up to two memory modules, and an expansion I/O module.

system bus

The hardware structure that interconnects the CPUs and memory modules. Data processed by the CPU is transferred throughout the system via the system bus.

system disk

The device on which the operating system resides.

TCP/IP

Transmission Control Protocol/Internet Protocol. A set of software communications protocols widely used in UNIX operating environments. TCP delivers data over a connection between applications on different computers on a network; IP controls how packets (units of data) are transferred between computers on a network.

test-directed diagnostics (TDDs)

An approach to diagnosing computer system problems whereby error data logged by diagnostic programs resident in read-only memory (RBDs) is analyzed to capture information about the problem.

thickwire

One-half inch, 50-Ohm coaxial cable that interconnects the components in many IEEE standard 802.3-compliant Ethernet networks.

ThinWire

Ethernet cabling and technology used for local distribution of data communications. ThinWire cabling is thinner than thickwire cabling.

Token Ring

A network that uses tokens to pass data sequentially. Each node on the network passes the token on to the node next to it.

twisted pair

A cable made by twisting together two insulated conductors that have no common covering.

uninterruptible power supply (UPS)

A battery-backup option that maintains AC power to a computer system if a power failure occurs.

warm bootstrap

A subset of the cold bootstrap operation. On AlphaGeneration systems, during a warm bootstrap, the console does not load PALcode, size memory, or initialize environment variables.

wide area network (WAN)

A high-speed network that connects a server to a distant host computer, PC, or other server, or that connects numerous computers in numerous distant locations.

Windows NT

"New technology" operating system owned by Microsoft, Inc. The AlphaServer systems currently support the Windows NT, OpenVMS, and Digital UNIX operating systems.

write back

A cache management technique in which data from a write operation to cache is written into main memory only when the data in cache must be overwritten.

write-enabled

Indicates a device onto which data can be written.

write-protected

Indicates a device onto which data cannot be written.

write through

A cache management technique in which data from a write operation is copied to both cache and main memory.

Index

How to Order Additional Documentation

Technical Support

If you need help deciding which documentation best meets your needs, call 800-DIGITAL (800-344-4825) and press 2 for technical assistance.

Electronic Orders

If you wish to place an order through your account at the Electronic Store, dial 800-234-1998, using a modem set to 2400- or 9600-baud. You must be using a VT terminal or terminal emulator set at 8 bits, no parity. If you need assistance using the Electronic Store, call 800-DIGITAL (800-344-4825) and ask for an Electronic Store specialist.

Telephone and Direct Mail Orders

From	Call	Write
U.S.A.	DECdirect Phone: 800-DIGITAL (800-344-4825) Fax: (603) 884-5597	Digital Equipment Corporation P.O. Box CS2008 Nashua, NH 03061
Puerto Rico	Phone: (809) 781-0505 Fax: (809) 749-8377	Digital Equipment Caribbean, Inc. 3 Digital Plaza, 1st Street Suite 200 Metro Office Park San Juan, Puerto Rico 00920
Canada	Phone: 800-267-6215 Fax: (613) 592-1946	Digital Equipment of Canada Ltd. 100 Herzberg Road Kanata, Ontario, Canada K2K 2A6 Attn: DECdirect Sales
International	————	Local Digital subsidiary or approved distributor
Internal Orders[1] (for software documentation)	DTN: 264-3030 (603) 884-3030 Fax: (603) 884-3960	U.S. Software Supply Business Digital Equipment Corporation 10 Cotton Road Nashua, NH 03063-1260
Internal Orders (for hardware documentation)	DTN: 264-3030 (603) 884-3030 Fax: (603) 884-3960	U.S. Software Supply Business Digital Equipment Corporation 10 Cotton Road Nashua, NH 03063-1260

[1] Call to request an Internal Software Order Form (EN–01740–07).

Reader’s Comments

Digital Alpha VME 2100
Service Guide
EK–DALPH–SG. A01

Your comments and suggestions help us improve the quality of our publications.

Thank you for your assistance.

I rate this manual’s:	Excellent	Good	Fair	Poor
Accuracy (product works as manual says)	□	□	□	□
Completeness (enough information)	□	□	□	□
Clarity (easy to understand)	□	□	□	□
Organization (structure of subject matter)	□	□	□	□
Figures (useful)	□	□	□	□
Examples (useful)	□	□	□	□
Index (ability to find topic)	□	□	□	□
Page layout (easy to find information)	□	□	□	□

I would like to see more/less ____________________

What I like best about this manual is ____________________

What I like least about this manual is ____________________

I found the following errors in this manual:

Page Description

Additional comments or suggestions to improve this manual:

For software manuals, please indicate which version of the software you are using: ______

Name/Title ____________________ Dept. __________

Company ____________________ Date __________

Mailing Address ____________________

____________________ Phone __________

Do Not Tear – Fold Here and Tape

digital™

No Postage
Necessary
If Mailed
in the
United States

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 33 MAYNARD MASS.

POSTAGE WILL BE PAID BY ADDRESSEE

DIGITAL EQUIPMENT CORPORATION
Shared Engineering Services
MLO5-5/E76
2 THOMPSON STREET
MAYNARD, MA 01754-1716

Do Not Tear – Fold Here