DECsystem 5900
CPU System Manual

Order Number EK–D590A–SM. A01

Digital Equipment Corporation
Maynard, Massachusetts

First Printing, June 1992

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.

S1921

This document was prepared using VAX DOCUMENT, Version 2.1.

Contents

Chapter 3 Troubleshooting Overview

Chapter 4 Troubleshooting FRUs

Chapter 5 Troubleshooting Tools

Appendix A Console Commands

Appendix B Base System Self-Test Commands and Error Messages

Appendix C CPU and System Registers

Appendix D Connector Pin Assignments

Appendix E ULTRIX System Exercisers

Index

Figures

Tables

Preface

Intended Audience

This guide is for Digital Service representatives who wish an in-depth view of DECsystem 5900 CPU operations and troubleshooting.

How To Use This Guide

This guide explains how the CPU works to control system operations and to interface to other devices. It also explains how to troubleshoot the system.

The CPU is the same as the one in the DECstation 5000 Model 240.

For an overview of the system hardware and its configurations, see Chapter 1, "System Overview."

For information about console mode, used for maintenance operations and operating mode, used for regular software operations, see Chapter 2, "Console Mode and Operating Mode."

For an overview of the tools that are used most often when troubleshooting the CPU and its peripherals, see Chapter 3, "Troubleshooting Overview."

For a description of the information available to help you identify failed FRUs, see Chapter 4, "Troubleshooting Information."

For a description of the tests and scripts used when troubleshooting, see Chapter 5, "Troubleshooting Tools."

For an explanation of console commands, see Appendix A, "Console Commands."

For an explanation of individual system module and memory module tests, see Appendix B, "Base System Self-Test Commands and Error Messages."

For information about connector pin assignments, see Appendix C, "CPU and System Registers."

For information about CPU and system registers, see Appendix D, "Connector Pin Assignments."

For information about ULTRIX system exercisers, see Appendix E, "ULTRIX System Exercisers."

Table 1: Conventions Used in This Guide

Convention	Use
`Monospace type`	`Anything that appears on your monitor screen is set in monospace type, like this.`
Boldface type	**Anything you are asked to type is set in boldface type, like this.**
Italic type	*Any part of a command that you replace with an actual value is set in italic type, like this.*
Note	Notes provide general information about the current topic.
Caution	Cautions provide information to prevent damage to equipment or software. Read these carefully.
Warning	Warnings provide information to prevent personal injury. Read these carefully.

Chapter 1
System Overview

This chapter provides an overview of the DECsystem 5900 hardware. This chapter discusses the following topics:

- Basic system hardware
- System hardware configurations
- Hardware options and peripherals

1.1 System Hardware Configurations

The DECsystem 5900 is a reduced instruction set computer (RISC) desktop system based on the MIPS R-3000 processor and designed to support the ULTRIX operating system.

The system is usually configured as a server.

1.2 External System Unit Connectors

The external system unit connectors on the back panel connect the workstation to external devices. The external system unit connectors are listed here and shown in Figure 1–1.

❶ Not used

❷ TURBOchannel Extender option slot

❸ TURBOchannel Extender option slot

❹ TURBOchannel Extender I/O (connected to ⓭)

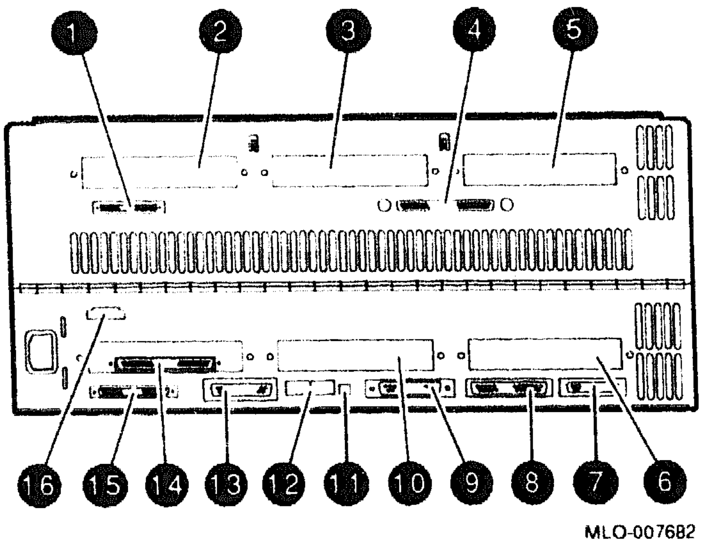
❺ TURBOchannel Extender option slot

❻ TURBOchannel option slot 2

❼ Not used

❽ Communications port

Figure 1–1: CPU Connectors



- 9 System console port
- 10 TURBOchannel slot 1
- 11 Halt switch
- 12 Diagnostic LEDs
- 13 Standard Ethernet
- 14 TURBOchannel Extender adapter (connected to 1)
- 15 System module SCSI port
- 16 Remote power sequence connector

1.3 DECsystem 5900 Functional Overview

The block diagram in Figure 1–2 shows the basic system block diagram from a functional perspective.

The DECsystem 5900 is made up of five main subsystems: memory, SCSI, Ethernet, TURBOchannel and the serial communication lines, and the clocking and interrupt handling logic to co-ordinate these functions as shown in Figure 1–2.

The system has four main Application Specific Integrated Circuits (ASIC) that make up much of the important logic circuits and control the movement of data in the system. These are the Memory Buffer (MB), Memory TURBOchannel (MT), Memory Subsystem (MS), and I/O Control (IOCTL).

The CPU daughter card mounts on the system module and contains the R3000A CPU/FPU, 64Kb data and instruction cache, a Memory Buffer (MB) ASIC and a Memory TURBOchannel (MT) ASIC.

The MB ASIC interfaces the CPU to the rest of the system, provides timing and control and cache RAMs interfacing.

The MT ASIC interfaces the CPU to memory and the TURBOchannel I/O interface. As a example, data can come in on the from the SCSI bus and DMA into memory by way of the MT ASIC without CPU intervention.

Located on the system module is the Memory Subsystem (MS) ASIC which provides interface to the memory SIMMs in the memory slots. Address and data pass through this ASIC. The MS ASIC also provides memory timing signals. Also on the system module is the SCSI, Ethernet, serial communication chip, run-time clock and supporting hardware for these devices.

These devices are controlled by the I/O Control (IOCTL) ASIC. The devices listed above are all on TURBOchannel slot 3, as the block diagram shows. The IOCTL ASIC also is the interface to memory and other parts of the system for TURBOchannel slots 0, 1, 2.

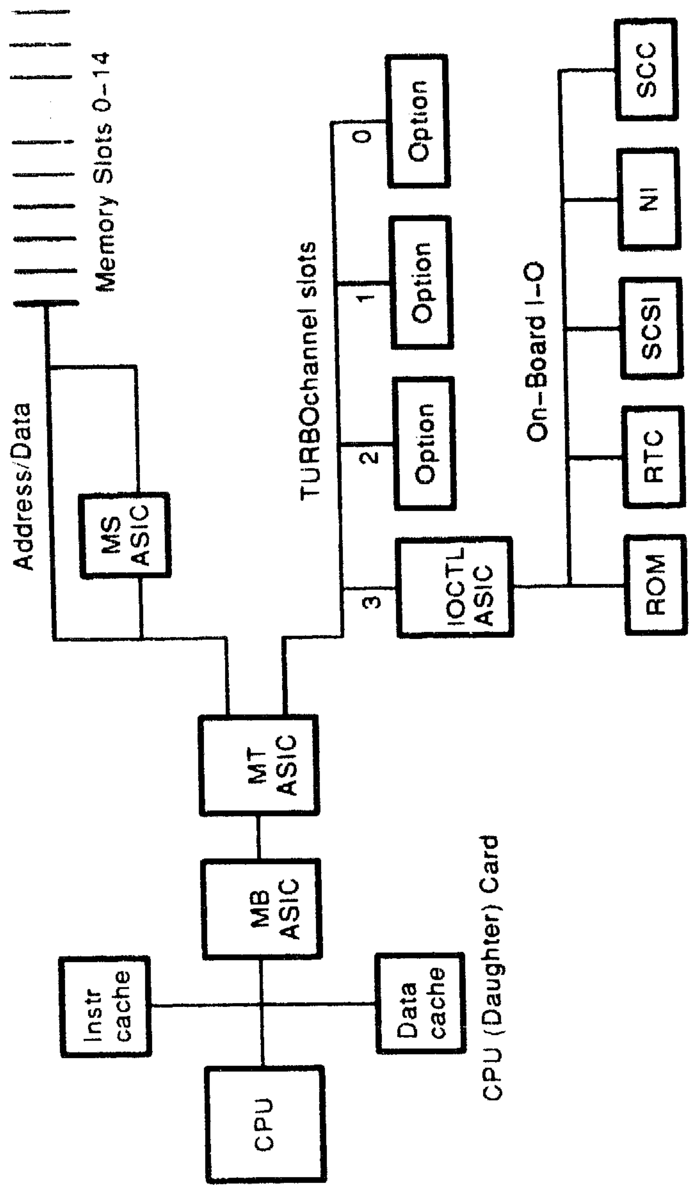
Since all I/O data passes through the IOCTL and control signals are provided by this ASIC, you must specify the TURBOchannel path when booting or testing a device on that slot. For example, cnfg 1 will only show you the devices on TURBOchannel slot 1. For the system to perform its power-up tests, it must first read the ROM by means of the IOCTL, MT and MB ASICS.

1.4 System Module Main Components

The system module is the largest printed circuit board in the CPU drawer, and is screwed directly to the floor of the CPU drawer. The system module supports the CPU module, Ethernet, SCSI bus, communication ports, memory modules, and up to three TURBOchannel options. Major elements on the system board include:

- 256-Kbyte power-up self test and bootstrap ROM
- System control and status registers and diagnostic LEDs
- RTC-based system clock and 50-byte (5-year) battery backed-up RAM

Figure 1–2: Block Diagram of CPU Control Paths



- SCC-based serial lines
- Two RS232 asynchronous serial comm ports
- Error address status register
- ECC error check/syndrome status register

- LANCE-based network interface for Ethernet
- Disk/tape interface for SCSI peripherals
- TURBOchannel I/O option connectors
- DMA for SCSI Ethernet, and two comm ports
- Halt switch

1.4.1 CPU Card Components

The CPU card is a daughter card held to the system module by four standoffs and a dual card-edge connector. It has the following functionality:

- R3000A MIPS processor (40 MHz) CPU/FPU
- Read/Write buffer
- 64-Kbyte instruction and data caches
- Processor interface
- Memory and TURBOchannel interface
- Clock and configuration logic

1.4.2 Memory Modules

Up to 14 32-Mbyte MS02–CA single in-line memory modules (SIMMs) may be connected to the system module, for a maximum total of 448 Mbytes. The standard DECsystem 5900 has a minimum of two 32-Mbyte MS02–CAs.

1.4.3 NVRAM Module

A 1-Mbyte NVRAM memory module can be installed in the last memory slot, slot 14. This is the only slot in which the NVRAM can be used, and is dedicated to the NVRAM only. The NVRAM module is the hardware that supports the PrestoServe NFS accelerator.

The NVRAM has two LEDs to indicate the status/condition of the on-board battery.

1.4.4 TURBOchannel and TURBOchannel Extender

TURBOchannel is an I/O interface. The system module contains three TURBOchannel option slots. One of these slots (0) is preconfigured with a adapter module that is used to connect a TURBOchannel extender. The remaining two slots (1 and 2) may be used for one dual or two single TURBOchannel option modules.

The TURBOchannel extender is a standard feature of the DECsystem 5900. The TCE is mounted in the hinged metal cover of the CPU drawer. The TURBOchannel extender allows a two- or three- slot TURBOchannel option module to be connected to the DECsystem 5900 and logically take up one TURBOchannel slot. This leaves two slots available for other TURBOchannel options. Without the TURBOchannel Extender, certain TURBOchannel options could use up all three system TURBOchannel slots, preventing other options from being installed.

TURBOchannel enables connection to TURBOchannel options such as:

PMAZ (single-ended SCSI)
PMAD (Ethernet)
DEFZA (FDDI)

As shown in Figure 1–1 there are three connections from the rear panel of the CPU drawer to optional TURBOchannel I/O devices. These are the TURBOchannel controller ports. One of them (slot 0) contains a TURBOchannel Extender adapter module and is connnected to the TURBOchannel Extender module (TCE). TCE provides a place to mount one-, two-, or three-slot option without using all of the system TURBOchannel slots.

NOTE: *Only one TURBOchannel option can be placed on the TURBOchannel Extender module. This module does not provide three logical slots but rather extends one slot outside of the system board.*

1.4.5 Internal Base System Module Connectors

The internal base system module connectors are the means by which the system's modular components, both standard and optional, are connected to the base system module and, through it, to each other.

- The CPU module connector connects the CPU module to the base system module.
- Fifteen memory module connectors provide the means for installing SIMMs and one optional NVRAM module.
- Three internal TURBOchannel option module connectors connect TURBOchannel option modules to the base system module. The connector closest to the power supply is referred to as logical slot number 2 in test commands and error messages. The middle connector is logical slot 1, and the connector furthest away from the power supply is logical slot 0.
- The power input connectors receive dc power from the power supply for all components in the system unit enclosure.

1.4.6 SCSI Drives

The base system module comes with a SCSI controller. In addition, up to three TURBOchannel SCSI controller option modules can be added to the system. Each SCSI controller can support up to seven drives.

1.4.7 TURBOchannel Option Modules

The TURBOchannel connectors on the base system module can support three TURBOchannel option modules. Any SCSI controller, Ethernet controller, or serial controller TURBOchannel option modules operate in addition to the equivalent functions on the base system module.

Chapter 2

Console Mode and Operating Mode

This chapter discusses the following topics:

- Console mode and operating mode
- Console prompts
- Password management
- System software startup and shutdown

2.1 Modes

The system operates in two modes: console mode and operating mode.

2.1.1 Console Mode

Most maintenance operations are conducted in console mode, including the following:

- Displaying hardware configurations (see Chapter 4 and Appendix A)
- Setting environment variables (see Chapters 3 and 5 and Appendix A)
- Running diagnostic tests and scripts (see Chapters 3 and 5 and Appendixes A and B)
- Booting the system software (see the"System Software Management" section in this chapter)

NOTE: *ULTRIX error logs cannot be accessed from console mode; they can be accessed only from operating mode.*

Console mode operations require that at least one RAM single in-line memory module (SIMM) be installed in slot 0 on the base system module.

2.1.1.1 Console prompts

When the normal console prompt (>>) is displayed, full console functionality is available, and you can use all console commands.

When the restricted console prompt (R>) is displayed, you can enter only the boot or passwd command without qualifiers.

Console commands, including boot and passwd, are discussed in Chapters 3 and Chapter 5 and Appendix A.

2.1.1.2 To enter console mode

This section lists the methods of entering console mode in order of recommended preference. Enter console mode in one of the following ways, depending on circumstances:

- If system software is running, shut down the system software. The system enters console mode automatically when system software is shut down. This is the most orderly way to enter console mode as it prevents corruption of the data.

  NOTE:: *Turning off the power while ULTRIX is running can corrupt data.*

- If autoboot is not enabled, turn off the system power, and then turn it on again. The system executes the power-up self-test sequence and then comes up in console mode, displaying one of the console prompts (>> or R>).

- If autoboot is enabled, you have to defeat autoboot to enter console mode by turning on the system power. Defeat autoboot in one of the following ways:

  — Turn off the system power, and then turn it on again.

    Watch the command line on the monitor. As each power-up self-test runs, the name of the test appears on the command line.

    When the screen does not display any test name and the cursor appears on a blank line, quickly press Ctrl-c.

    The system aborts the boot process and comes up in console mode, displaying one of the console prompts (>> or R>)

  — Set the `setenv` variable to `setenv haltaction -h`.

  — When finished, set the `setenv` variable to `setenv haltaction -b`.

If the R> prompt is displayed, you can use only the boot and passwd commands until you enter the password.

2.1.1.3 Halt button

The Halt button on the rear panel interrupts the processor. The data in the memory is preserved.

2.1.2 Operating Mode

In operating mode, the system displays the ULTRIX prompt. Operating mode is used for regular software operation.

For maintenance purposes, operating mode is used to access ULTRIX error logs.

2.1.2.1 To enter operating mode

The system enters operating mode in one of two ways: automatically (autoboot) or manually from console mode.

If autoboot is enabled, the system executes the boot command immediately after the power-up self-test. The system goes directly to operating mode without displaying the console prompt. Autoboot is enabled when the haltaction environment variable has been set to b and the boot environment variable has been set to a meaningful bootpath. The boot and setenv commands and the environment variables are discussed in Appendix A.

Procedures for entering operating mode manually from either console prompt are described in Section 2.2.1.

2.2 System Software Management

The following system software (ULTRIX) operations are significant to the hardware maintenance process:

- Starting up (booting) system software
- Shutting down system software
- Accessing ULTRIX error logs

2.2.1 To Boot System Software

1. At either console prompt `>>` or `R>`, enter **boot** and press Return. The boot process takes several minutes.
2. If the system displays the ULTRIX prompt (#) before the `login:` prompt appears, the system has stopped at single-user mode instead of multiuser mode. To move on to multiuser mode, press `Ctrl-d` to continue the boot operation. When the system displays the `login:` prompt, the system software has started successfully. The system probably stopped at single-user mode because the bootpath is set for

single user mode or because of disk corruption. See the "setenv Command" section in Appendix A for information about how to set the bootpath for multiuser mode. If the problem persists, clean the disks using the fsck function.

3. If the system displays a console prompt (>> or R>), the boot failed. Proceed as follows:

   a. If the system displays an error message, see the "Console Command Error Messages" section in Appendix A .

   b. If the the system displays the restricted prompt (R>), enter **passwd** and press Return. At the pwd: prompt, enter the password and press Return. The system displays the console prompt (>>). If you cannot enter the password, see the *DECsystem 5900 Service Guide*.

   c. At the console prompt (>>), enter **printenv** and press Return to display the environment variables table.

   d. Use the setenv command to set the boot environment variable to a device or to the network that contains the system software that you want to boot. See the "boot Command" section in Appendix A .

   e. Reenter the boot command to boot the system.

2.2.2 To Shut Down System Software

If the system is running ULTRIX software, shut down the software before you perform hardware maintenance.

At the ULTRIX prompt (#), enter

/etc/shutdown -h (**now** | *hhmm* | +*n*)

and press Return.

You must include one of the parameters shown in parentheses to tell the system when to shut down.

- Specify the now parameter to shut down the software immediately.
- Specify *hhmm* to shut down the software at a specific hour and minute.
  - Replace *hh* with the hour to begin the shutdown.
  - Replace *mm* with the minute to begin the shutdown.
- Specify +*n* to shut down the software in a specified number of minutes. Replace n with the number of minutes until shutdown begins.

The system displays a console prompt (>>) or (R>) when shutdown is complete.

2.2.3 To Access ULTRIX Error Logs

At the ULTRIX prompt (#), enter

/etc/uerf (-R | more)

and press Return. For information about interpreting the ULTRIX error logs, see the "ULTRIX Error Logs" section in Chapter 4.

Chapter 3
Troubleshooting Overview

3.1 Introduction

This chapter provides a brief overview of the tools that are used most often when troubleshooting the CPU and its peripherals.

The field service engineer will solve each problem differently as logic dictates. This chapter is an overview of the most often-used troubleshooting tools and techniques. Each of the tools and techniques mentioned is covered in detail in its own section.

- Chapter 4 discusses the information that the field service engineer uses to identify failed field-replaceable units (FRUs).
- Chapter 5 discusses the tools that the engineer uses to test the system and its components.

In general, these are the questions that the Digital Services engineer deals with when working on a system:

- What malfunction does the user report?
- What malfunction does the field engineer observe?
- Have the proper procedures been followed?
- Has the system run properly in the past or is it a new system?
- Are the cables and connectors in order?
- Is power getting to the system and its components?
- Does the screen work?
- When the power-up self-test sequence runs, do error messages appear on the screen or on the diagnostic LED array on the rear panel of the system unit?
- What useful information does the cnfg display provide?
- Are the environment variables set properly?
- What useful information can the tests and scripts provide?

- Is the software version appropriate? If this problem is suspected, check with the technical support group at Digital or the module vendor for further information.

3.2 Power

If the green LED on the front of the system unit or any of the mass storage drawers doesn't light up, the first priority is to get power to the device.

3.3 Self-Tests

When the self-test sequence runs,

- If the system presents no error codes and displays the console prompt (>>) or boots up, go on.
- If an error code is displayed on the diagnostic LED array but not on the screen, use the LED error code to troubleshoot. See the "LED Displays" section in Chapter 4.
- If one or more error messages appears on the screen, use the error messages to troubleshoot. See the "Error Messages" section in Chapter 4 and the "Power-Up Self-Tests" section in Chapter 5.
- When the console prompt (>>) appears, you can use the console tests and utilities to get more information. See Chapter 5.

3.4 Configuration Displays

To see the configuration overview display, at the console prompt (>>) enter

cnfg

and press Return.

To view the detailed configuration information for one module, enter

cnfg *slot_number*

and press Return. Replace *slot_number* with the number of the slot where the module is installed. See the "cnfg Command" section in Appendix A.

Look for the following information:

- Does all of the installed hardware appear in the configuration display?
- Does the right amount of memory appear?
- Are the SCSI IDs correct?

- Is the firmware version appropriate? If this problem is suspected, check with the technical support group at Digital or the module vendor for further information.

3.5 Environment Variables

Environment variables are set as follows:

At the console prompt (>>) enter

printenv

and press Return. See the "printenv Command" and "setenv Command" sections in Appendix A.

Look for the following information:

- Does the boot variable refer to the correct drive and is the drive working? See the "boot Command" section in Appendix A.
- Is the haltaction variable set as desired, either to autoboot or to stop in console mode.?

3.6 Tests and Scripts

The base system and the TURBOchannel option modules contain tests and scripts that can be used to test functions and components singly or in combination.

3.6.1 Tests

To view the tests that are available for a module, at the console prompt (>>) enter

t *slot_number*/?

and press Return. Replace *slot_number* with the number of the slot where the module to be tested is installed.

Table B–1, "Base System Module Tests and Utilities", lists all of the tests for the base system module and indicates the function assessed by each test.

3.6.2 Scripts

To view the scripts available for a module, at the console prompt (>>) enter

ls *slot_number*

and press Return. Replace *slot_number* with the slot where the module to be tested is installed.

To view the contents of a script, at the console prompt (>>) enter

cat *script_name*

and press Return. Replace *script_name* with the name of the script.

You can write your own script to assemble a set of tests and scripts appropriate to a given troubleshooting situation. See the "To Create a Test Script" section in Chapter 5.

Chapter 4

Troubleshooting FRUs

This chapter describes the information available to help you identify failed FRUs. The types of troubleshooting information are as follows:

- LED displays
- Configuration displays
- Error messages
- Addresses
- ULTRIX error logs
- Registers

Some of the information, such as exception messages and power-up self-test error messages, is displayed automatically. Other information, such as configuration displays, test error messages, ULTRIX error logs, and registers, must be specifically generated or accessed by the engineer. ULTRIX error logs are accessible only in operating mode. All of the other types of configuration information are accessible only in console mode. See Chapter 2 for information about console and operating modes.

4.1 LED Displays

The following three LED displays provide information about malfunctions:

- Diagnostic LED array
- CPU module LEDs
- Power supply (DCOK) LED

4.1.1 Diagnostic LED Array

The LEDs at the center rear of the CPU drawer indicate that tests are proceeding. This is useful when the system cannot display error messages on the console device.

4.1.2 CPU Module LEDs

A pair of LEDs on the CPU module light up when certain power-up events occur. When the power-up self-test fails to complete, the status of the CPU module LEDs implies the following:

- If neither LED lights up, the CPU module is likely faulty.
- If only one LED lights up, the base system module is likely faulty.
- If both LEDs light up, the CPU and base system modules have completed basic communication operations with each other.

4.1.3 Drawer LEDs

The green LED on the on/off switch at the front of each drawer (and at the rear switch of some versions of the mass storage drawer) indicates when power is on in that drawer.

4.2 Configuration Displays

The configuration displays show what devices are installed in the system. When hardware does not show up on the configuration display, or shows up incorrectly, this fact can be useful for troubleshooting. You can also use configuration displays to obtain the following information about the components:

- The amount of RAM memory installed on a board
- Whether an NVRAM module is installed
- The Ethernet address of an Ethernet controller
- The SCSI ID and Device Type of SCSI devices

See the "cnfg Command" section in Appendix A.

You can request configuration information in either of two forms:

- You can request a configuration overview, which provides basic information about the hardware installed in all of the TURBOchannel slots.
- You can request detailed information about the hardware in one TURBOchannel slot.

4.2.1 Configuration Overview

For the configuration overview, at the console prompt (>>) enter

cnfg

and press Return. The following is a typical configuration overview display:

```
3:    KN03-AA    DEC    V5.0a    TCF    ( 64MB,     1MB NVRAM)
                                        (enet: 08-00-2b-24-5b-82)
                                        (scsi = 7)

|        |        |       |       |     (Installed RAM)
|        |        |       |       |     (Ethernet address)
|        |        |       |       |     (SCSI ID)
|        |        |       |       |
|        |        |       |     Firmware type
|        |        |    Firmware version
|        |     Module vendor
|      Module name (base system module)
TURBOchannel slot number
```

- `64MB, 1MB NVRAM`
  64 megabytes of RAM and 1 megabyte of NVRAM is installed on the base system module.
- `enet: 08-00-2b-64-5b-82`
  The Ethernet address of the base system Ethernet controller.
- `scsi = 7`
  The SCSI ID of the base system SCSI controller.

4.2.2 Detailed Configuration

For detailed information about the hardware in one TURBOchannel slot, at the console prompt (>>) enter

cnfg *slot_number*

and press Return. Replace *slot_number* with the slot number of the module for which you want configuration information.

For example, for detailed information about the base system module hardware, at the console prompt (>>) enter

cnfg 3

and press Return. The following is a typical detailed configuration display for the base system module:

```
3: KN03-AA   DEC   V5.0a   TCF0   ( 96MB,     1MB NVRAM)
                                  (enet: 08-00-2b-64-5b-82)
                                  (scsi = 7)
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
                        DEV   PID             VID   REV    SCSI DEV
                        ===   ============    ===   ===    ========
                        rz1   RZ58   (c) DEC  DEC   nnnn   DIR
                        rz2   RRD42  (c) DEC  DEC   nnnn   CD-ROM

                  dcache   ( 64 KB), icache   ( 64 KB)
                  mem( 0): a0000000:a1ffffff  (  32 MB)
                  mem( 1): a2000000:a3ffffff  (  32 MB)
                  mem( 2): a4000000:a5ffffff  (  32 MB)
                  mem(14): a1800000:a18fffff  (  1 MB)   Presto-NVR

                  mem(14): clean, batt ok, armed
```

This detailed configuration display example provides the following information in addition to the configuration overview:

- The three SCSI drives connected to the base system module are as follows:
  - `rz1 RZ58 (c) DEC DEC nnnn DIR:`
    An RZ58 drive with SCSI ID 1, manufactured by Digital Equipment Corporation, is using firmware version nnnn and is a hard disk drive (DIR).
  - `rz2 RRD42 (c) DEC DEC nnnn CD-ROM:`
    An RRD42 optical medium with SCSI ID 2, manufactured by Digital Equipment Corporation, is using firmware version nnnn and is a CD-ROM.
- `dcache ( 64 KB), icache ( 64 KB):`
  The base system module data cache is 64 kilobytes. The base system module instruction cache is 64 kilobytes.
- The RAM configuration is as follows:
  - `mem( 0): a0000000:a1ffffff ( 32 MB)`
    32 megabytes of memory in memory slot 0 are assigned memory addresses a0000000 to a1ffffff .
  - `mem( 1): a2000000:a3ffffff ( 32 MB)`
    32 megabytes of memory in memory slot 1 are assigned memory addresses a2000000 to a3ffffff.
  - `mem( 2): a4000000:a5ffffff ( 32 MB)`
    32 megabytes of memory in memory slot 2 are assigned memory addresses a4000000 to a5ffffff.

— `mem( 14): a1800000:a18ffff ( 1 MB)`
Memory slot 14 contains a 1-megabyte NVRAM module. Addresses a1000000 to a18fffff are assigned to the NVRAM module.

— `mem(14): clean, batt ok, armed:`
The memory in the NVRAM module is clean, the battery is okay, and the battery in not turned on.

4.3 Error Messages

An error message can be a test error message or a console exception message. Test error messages are displayed when an automatic or user-initiated test fails. Console exception messages are automatically displayed when console operations fail.

This section describes the following error message types:

- Test error messages
- Console exception messages
- Memory test error messages

4.3.1 Test Error Messages

When a test fails, the message appears on the screen in the following format:

```
?TFL slot_number/test_name
(n:description)
[module].
```

?TFL	Identifies a test error message.
slot_number	Identifies the module that reported the error.
test_name	Identifies the test that failed.
n	Indicates which part of the test failed.
description	Describes the failure; the message may include an address.
module	Indicates the module identification number.

For an explanation of system and memory module test error messages , see Appendix B. For information about other error messages, see Appendix A. For an explanation of TURBOchannel option module error messages, refer to the *TURBOchannel Maintenance Guide*.

This is a typical error message:

```
?TFL 3/scsi/cntl (3: cnt xfr) [KN03-AA]
```

This error message states that the KN03-AA module in slot number 3, the base system module, failed the SCSI controller test. The explanation of the SCSI controller test in Appendix B states that the message `(3: cnt xfr)` means that the read and write operation reported a mismatch. Table 4–1 lists the base system tests and the corrective action indicated when each test is listed in a test error message.

Table 4–1: Base System Test Error Messages

Error Message	Component	Corrective Action
cache/data cache/fill cache/isol cache/reload cache/seg fpu	CPU module	Replace the CPU module. If the problem persists, replace the system module.
ecc/cor mem mem/float10	Memory modules	See the appropriate test section in Appendix B.
mem/select	Memory and system module	Replace the memory module that the test identifies. If the problem persists, replace the system module.
misc/halt	System module	Replace the system module.
misc/pstemp	Power supply	See the misc/pstemp test section in Appendix B.
misc/wbpart	Memory modules	See the misc/wbpart test section in Appendix B.
ni/cllsn ni/common ni/crc ni/cntrs ni/dma1 ni/dma2 ni/esar ni/ext-lb ni/int ni/int-lb ni/m-cst ni/promisc ni/regs	Base system Ethernet controller	See the appropriate test section in Appendix B.
prcache prcache/arm prcache/clear prcache/unarm	NVRAM module	See the appropriate test section in Appendix B.

Table 4–1 (Cont.): Base System Test Error Messages

Error Message	Component	Corrective Action
rtc/nvr rtc/period rtc/regs rtc/time	System module	Replace the system module.
scc/access scc/dma scc/init scc/io scc/pins scc/tx-rx	Serial line controllers and devices attached to them	See the appropriate test section in Appendix B.
scsi/cntl scsi/sdiag scsi/target	Base system SCSI controller or device	See the appropriate test section in Appendix B.
tlb/prb tlb/reg	CPU module	Replace the CPU module.

4.3.2 Memory Test Error Messages

When a memory test detects an error, the message appears on the screen in the following format:

?TFL 3/mem (*n*: board *xx*, MBE = *yy*, SBE = *zz*)

?TFL 3/mem	Indicates that a memory test failed.
n	Represents the number of the subtest that failed.
xx	Represents the memory slot where the faulty board is installed.
yy	Represents the number of multiple-bit errors that occurred.
zz	Represents the number of single-bit errors that occurred.

This is a typical memory test error message:

```
?TFL:3/mem (1: board 3, MBE = 25, SBE = 6))
```

In this example:

3/mem	Indicates that the mem test failed.
1:	Indicates that subtest number 1 failed.
board 3	Indicates that the SIMM in slot 3 is faulty.
MBE = 25	Indicates that 25 multiple bit errors occurred.
SBE = 6	Indicates that 6 single bit errors occurred.

4.3.3 Console Exception Messages

When a console operation fails, the system displays a console exception message. When a console exception message appears, first verify that any command and address that you entered are valid. If you are sure the command and address are correct but the console exception still occurs, interpret the message to determine what caused the exception. For information about the registers, see Appendix C.

A console exception message can be recognized by the first line, which always begins with the characters ? PC:. A console exception message includes some combination of the following entries:

```
? PC:  address
? CR:  cause
? SR:  status
? VA:  virtual address
? ER:  error address
? CK:  error syndrome
```

- *address* represents the address of the exception instruction.
- *cause* represents the value in the cause register.
- *status* represents the contents of the status register.
- *virtual address* represents the virtual address of the exception.
- *error address* represents the contents of the error address register.
- *error syndrome* represents the value in the error syndrome register.

The following example shows a typical value for each of the possible entries of a console exception message. In each entry, the information in brackets is the decoded version of the hexadecimal value that precedes it.

```
? PC: 0xbfc0d0d <vtr=NRML>
? CR: 0x210c <CE=0,IP6,EXC=DBE>
? SR: 0x30080000 <CU1,CU0,CM,IPL=8>
? VA: 0x0
? ER: 0xd0800006 <VALID,CPU,ECCERR,ADR=2000018>
? CK: 0x8c18c321
<VLDHI,CHKHI=C,SYNHI=18,VLDLO,CHKLO=43,SYNLO=21>
```

4.4 Addresses

Addresses of various types appear in error and exception messages. These addresses indicate the location of the malfunction. You use addresses in test commands to indicate which module or memory location the test is to address.

This section describes the following types of addresses:

- Slot numbers
- Memory addresses
- Hardware physical addresses

4.4.1 Slot Numbers

Test commands and error messages include slot numbers that identify the hardware to which the test command or error message refers, as shown in Table 4–2.

Table 4–2: Slot Numbers in Commands and Messages

Slot	Hardware Identified
0	Option module in slot 0 (on left side, viewing from rear)
1	Option module in slot 1 (middle option slot)
2	Option module in slot 2 (on right side, viewing from rear)
3	Base system hardware, including — System module — CPU module — SIMMs — NVRAM — Serial communications controller — Base system SCSI controller — Base system Ethernet controller

4.4.2 Memory Addresses

When a memory error occurs, the error message contains the address of the error. You can identify the faulty SIMM by the address.

Addresses can appear in error messages in several formats, but you must use the kseg1 format to specify addresses in console commands. kseg1 format refers to uncached, unmapped address space. In kseg1 format, the uppermost three bits of the address are always 101 and the hexadecimal form of the address always begins with an A or a B. For example, if an address is listed in an error message as 0x04040404, you would use 0xA4040404 to specify that address in a console command. If an address is listed in an error message as 0x14040404, you would use 0xB4040404 to specify that address in a console command.

Table 4–3 lists the memory addresses in kseg1 format by slot number for 32-Mbyte memory modules.

Table 4–3: Memory Module Address Ranges

Memory Slot	32-Mbyte Module Address Ranges
0	0xA0000000 to 0xA1FFFFFF
1	0xA2000000 to 0xA3FFFFFF
2	0xA4000000 to 0xA5FFFFFF
3	0xA6000000 to 0xA7FFFFFF
4	0xA8000000 to 0xA9FFFFFF
5	0xAA000000 to 0xABFFFFFF
6	0xAC000000 to 0xADFFFFFF
7	0xAE000000 to 0xAFFFFFFF
8	0xB0000000 to 0xB1FFFFFF
9	0xB2000000 to 0xB3FFFFFF
10	0xB4000000 to 0xB5FFFFFF
11	0xB6000000 to 0xB7FFFFFF
12	0xB8000000 to 0xB9FFFFFF
13	0xBA000000 to 0xBBFFFFFF
14	0xBC000000 to 0xBDFFFFFF
Reserved	0xA7800000

4.4.3 Hardware Physical Addresses

The hardware addresses in Table 4–4 appear in ULTRIX error logs.

Table 4–4: Hardware Physical Addresses

Physical Address Range	Indicated Hardware
0x00000000 to 0x1DFFFFFF	RAM
0x1E000000 to 0x1E7FFFFF	TURBOchannel slot 0
0x1E800000 to 0x1EFFFFFF	TURBOchannel slot 1
0x1F000000 to 0x1F7FFFFF	TURBOchannel slot 2
0x1F800000 to 0x1FFFFFFF	Slot 3: Base system module
The following addresses are included in the system module address range:	
0x1F800000 to 0x1F83FFFF	System ROM
0x1F840000 to 0x1F87FFFF	Input/output control (IOCTL) registers and direct memory access (DMA) pointers
0x1F880000 to 0x1F8BFFFF	Ethernet address PROM and EEPROM
0x1F8C0000 to 0x1F8FFFFF	Ethernet interface
0x1F900000 to 0x1F93FFFF	Serial communication chip (SCC)(0) registers
0x1F940000 to 0x1F97FFFF	Reserved
0x1F980000 to 0x1F9BFFFF	SCC(1) registers
0x1F9C0000 to 0x1F9FFFFF	Reserved
0x1FA00000 to 0x1FA3FFFF	Real-time clock
0x1FA40000 to 0x1FA7FFFF	Error address (EA) register (0x1FA40000)
0x1FA80000 to 0x1FABFFFF	Error syndrome (ES) register (1FA80000)
0x1FAC0000 to 0x1FAFFFFF	Control/status (CS) register (0xFAC0000)
0x1FB00000 to 0x1FB3FFFF	SCSI interface
0x1FB40000 to 0x1FB7FFFF	Reserved
0x1FB80000 to 0x1FBBFFFF	SCSI DMA
0x1FBC0000 to 0x1FBFFFFF	Reserved
0x1FC00000 to 0x1FC3FFFF	Boot ROM
0x1FC40000 to 0x1FFFFFFF	Reserved
0x20000000 to 0x3FFFFFFF	TURBOchannel slot 0
0x40000000 to 0x5FFFFFFF	TURBOchannel slot 1
0x60000000 to 0x7FFFFFFF	TURBOchannel slot 2
0x80000000 to 0xFFFFFFFF	Reserved

4.5 ULTRIX Error Logs

The system records events and errors in the ULTRIX error logs. Use the memory error logs and the error and status register error logs to troubleshoot intermittent problems. This section describes ULTRIX error log formats and error log items that are useful for troubleshooting.

The ULTRIX error logs are not the same as the test error logs that appear when you use the erl command from the console prompt. A test error log is a record of errors reported by tests run in console mode.

4.5.1 Examining Error Logs

You must be running ULTRIX to examine error logs. At the ULTRIX prompt (#) enter

/etc/uerf -R | more

and press Return. A full display of ULTRIX error logs, with the newest error logs first, appears on the monitor.

For information about running ULTRIX, see the "System Software Management" section in Chapter 2. Information about the uerf command in the ULTRIX man facility can be obtained by entering **man uerf** at the ULTRIX prompt (#).

4.5.2 ULTRIX Error Log Format

The first part of each ULTRIX error log describes the type of error and system conditions in effect when the error occurred. The format of the first part is the same for all ULTRIX error logs, regardless of the event type.

The second part of each log provides specific information about the error and its location. In the second part, the information available for troubleshooting varies according to the event type.

The first part of all ULTRIX error logs is similar to this example:

```
****************** ENTRY    6.  ******************
```

```
----- EVENT INFORMATION -----
EVENT CLASS                            OPERATIONAL EVENT
OS EVENT TYPE                  250.    ASCII MSG
SEQUENCE NUMBER                  5.
OPERATING SYSTEM                       ULTRIX 32
OCCURRED/LOGGED ON                     Mon Nov 11  10:39:27  1992 EST
OCCURRED ON SYSTEM                     GRANITE
SYSTEM ID                 x82040230    HW REV: x30
                                       FW REV: x2
                                       CPU TYPE: R2000A/R3000
PROCESSOR TYPE                         KN03
MESSAGE                          Error count on memory module 0 reached
                                       _2048, resetting count to zero.
```

- EVENT CLASS indicates the category of the error. The two event class categories are operational events and error events.
  - Operational events are changes in system operation that are not errors.
  - Error events are actual errors in system operation.
- OS EVENT TYPE describes the type of error or event recorded in the log. Table 4–5 lists the operating system event types and their codes. For information about memory error logs and error and status register error logs, see the "Ultrix Error Log Event Types" and "Memory Error Logs" sections in this chapter.
- SEQUENCE NUMBER indicates the order in which the system logged the event.
- OPERATING SYSTEM indicates the system's version of ULTRIX.
- OCCURRED/LOGGED ON indicates when the error occurred.
- OCCURRED ON SYSTEM identifies the system that reported the error.
- SYSTEM ID includes the following listings:
  - The first number is the system ID.
  - HW REV indicates the system hardware revision number.
  - FW REV indicates the system firmware revision number.
  - CPU TYPE indicates the type of CPU installed in the system.
  - PROCESSOR TYPE indicates the type of processor chip that the system uses.
- The MESSAGE field provides information about the event or error.

4.5.3 ULTRIX Error Log Event Types

The second line of each error log indicates the code number and event type of the error. Table 4–5 lists the error log event types.

Table 4–5: Error Log Event Types

Code	Event Type
100	Machine check
101	Memory error
102	Disk error
103	Tape error
104	Device controller error
105	Adapter error
106	Bus error
107	Stray interrupt
108	Asynchronous write error
109	Exception or fault
113	CPU error and status information
130	Error and status register
200	Panic (bug check)
250	Informational ASCII message
251	Operational message
300	System startup message
310	Time change message
350	Diagnostic information

The information in the second part of an error log varies according to the event type listed on line 2 of the first part of the error log.

For a detailed explanation of other error logs, refer to the ULTRIX documentation for the uerf function or the documentation for the device that the error log discusses.

4.5.4 Memory Error Logs

The two examples in this section are two sequential ULTRIX error log entries that are related to each other. The two entries were generated when a correctable single-bit error occurred in the SIMM in slot 0. ENTRY 6 occurred within 1 second after ENTRY 5.

```
****************** ENTRY   6. ******************
----- EVENT INFORMATION -----
EVENT CLASS                          OPERATIONAL EVENT
OS EVENT TYPE               250.     ASCII MSG
SEQUENCE NUMBER               5.
OPERATING SYSTEM                     ULTRIX 32
OCCURRED/LOGGED ON                   Mon Nov 11  10:39:27  1991 PST
OCCURRED ON SYSTEM                   GRANITE
SYSTEM ID              x82040230     HW REV: x30
                                     FW REV: x2
                                     CPU TYPE: R2000A/R3000
PROCESSOR TYPE                       KN03
MESSAGE                     Error count on memory module 0 reached
                                     _2048, resetting count to zero.

****************** ENTRY 5.  ******************
----------- EVENT INFORMATION  ---------------
EVENT CLASS                     ERROR EVENT
OS EVENT TYPE             101. MEMORY ERROR
SEQUENCE NUMBER             4.
OPERATING SYSTEM                ULTRIX 32
OCCURRED/LOGGED ON                 Mon Nov 11  10:39:27  1991 PST
OCCURRED ON SYSTEM              GRANITE
SYSTEM ID           x82040230 HW REV: x30
                          FW REV: x2
                          CPU TYPE: R2000A/R3000
PROCESSOR TYPE                  KN03
-------------- UNIT INFORMATION -------------
UNIT CLASS                      MEMORY
UNIT TYPE             MS02 MEMORY
ERROR SYNDROME                  MEMORY CRD ERROR
-------------- KN03 MEMORY REGISTERS -------
EPC                        x800AFA3C
MEMORY CSR         x00006400   CHECK VALUE x0
                          32 MB MEM MODULES
                          ECC ERROR CORRECTION ENABLED
PHYSICAL ERROR ADDR        x010205F8
CHECK SYNDROME                x00308CB4    SYND BITS x34
                                           SINGLE BIT ERROR
                                           CHECK BITS xC
                                           MODULE NUM. x0
                                           ERROR COUNT 3.
                                           INVALID PC MEMINTR
```

A troubleshooter would analyze the error logs in the preceding examples as discussed here. See Appendix C for detailed information about memory registers.

- The MESSAGE field of ENTRY 6 indicates that more than 2048 single-bit ECC errors have occurred on the memory module in memory slot 0 and the counter has been reset to zero. Since the memory error correction feature corrects single bit errors, this is an operational event, not strictly an error.
- ENTRY 5 reports the actual single-bit error that overflowed the counter, causing it to be reset. The information under the KN03 MEMORY REGISTERS heading is useful to the troubleshooter:
  - SINGLE BIT ERROR indicates that a single-bit correctable error occurred.
  - MODULE NUM x0 indicates that the error occurred on the module in slot 0.
  - The PHYSICAL ERROR ADDR field indicates the error address.
  - The value 32 MB MEM MODULES in the MEMORY CSR field indicates the size of the memory modules.

ECC memory is designed so that occasional single-bit errors can occur and correction will take place. If occasional errors occur on a module, the module should not be replaced. But if a particular memory module records errors frequently, the module should be replaced. The memory error log example in this section describes a multiple-bit uncorrectable error. The module where the error occurred must be changed.

```
******************* ENTRY   193. *******************

----- EVENT INFORMATION -----

EVENT CLASS                              ERROR EVENT
OS EVENT TYPE                   101.     MEMORY ERROR
SEQUENCE NUMBER                   7.
OPERATING SYSTEM                         ULTRIX 32
OCCURRED/LOGGED ON                       Tue Jan  7 10:52:18 1992
PST
OCCURRED ON SYSTEM                       csselab2
SYSTEM ID                  x82040230     HW REV: x30
                                         FW REV: x2
                                         CPU TYPE: R2000A/R3000
PROCESSOR TYPE                           KN03

----- UNIT INFORMATION -----
```

```
UNIT CLASS                                  MEMORY
UNIT TYPE                                   MS02 MEMORY
ERROR SYNDROME                              MEMORY RDS ERROR

----- KN03 MEMORY REGISTERS -----

EPC                         x8011995C
MEMORY CSR                  x00006400   CHECK VALUE x0
                                        32 MB MEM MODULES
                                        ECC ERROR CORRECTION ENABLED
PHYSICAL ERROR ADDR         x00FD5ECC
CHECK SYNDROME              x800080B5   SYND BITS x35
                                        SINGLE BIT ERROR
                                        CHECK BITS x0
                                        MODULE NUM. x0
                                        ERROR COUNT 0.
```

- The ERROR SYNDROME field describes the error. The value in that field (MEMORY RDS ERROR) indicates that a multi-bit uncorrectable error occurred.
- The information under the KN03 MEMORY REGISTERS heading provides the following useful information:
  - The value NUM. x0 in the fourth line of the CHECK SYNDROME field indicates that the error occurred on the module in slot 0.
  - The PHYSICAL ERROR ADDRESS field indicates the error address.
  - The value 32 MB MEM MODULES in the second line of the MEMORY CSR filed indicates the size of the memory modules.

Replace the indicated memory module. Multi-bit errors are not correctable, and will cause processes and the system to crash.

4.6 Registers

The system automatically displays CPU register information in the console exception message when console exception exception occurs. To access system registers, from the console prompt (>>) enter

e *console_address*

and press Return.

Replace *console_address* with the address of the register that you want to examine. Use the kseg1 format for the address.

For information about the kseg1 format, see the "Memory Addresses" section in this chapter. For complete register information, see Appendix C. For information about the e command, see the "e Command" section in Appendix A.

4.7 For Further Information

For an explanation of other error logs, refer to the ULTRIX documentation for the uerf function.

For an explanation of error logs for SCSI devices, refer to the documentation for the device described in the error log.

Chapter 5

Troubleshooting Tools

This chapter discusses the system troubleshooting tools. It explains how to

- Run tests
- Use test scripts

5.1 Console Mode

You have to be in console mode to perform maintenance operations, including the following:

- Run diagnostic tests
- Read error messages
- Set environment variables
- Display hardware configurations

See the "Console Mode" section in Chapter 2

NOTE: *You have to be in operating mode to use ULTRIX error logs.*

5.2 Tests

The read-only memory (ROMs) on the base system module and on the TURBOchannel option modules contain numerous tests that verify the functions of the system. Tests can be used in the following ways to check system hardware operation:

- The automatic power-up self-test scripts run a comprehensive set of individual tests on the system and option module hardware.
- You can run individual tests in console mode to test specific system and option module functions.
- You can run one of several prepared scripts or create a script of your own, containing any set of tests that you find appropriate.

5.2.1 Slot Numbers in Test Commands and Error Messages

Test commands and error messages use slot numbers to identify the hardware to which the command or message refers.

- Slot 3 always refers to the base system hardware, which includes the following:
  - System module
  - CPU module
  - Memory modules (SIMMs and NVRAM)
  - Base system Ethernet controller
  - Base system SCSI controller
  - Serial line controllers
- Slot 0 refers to the TURBOchannel option slot on the left side, viewing from the back.
- Slot 1 refers to the middle TURBOchannel option slot.
- Slot 2 refers to the TURBOchannel option slot on the right side, viewing from the back.

5.3 Power-Up Self-Tests

When you turn on the system power, the system automatically runs a power-up self-test script. The monitor and the diagnostic LED array report any errors the power-up self-tests detect. Self-test error codes are discussed in the "Error Messages" section in Chapter 4 and in Appendix B.

You can specify a quick or a thorough power-up self-test script to run when the system powers up.

- The quick script, usually specified for normal power-up, is a limited script that allows the system to boot quickly
- The thorough script runs an extensive check of system hardware. The thorough script is most useful for field service troubleshooting.

To select a power-up self-test script, use the setenv command to set the testaction environment variable. Enter

setenv testaction (q | t)

and press Return. The vertical bar (|) means that you choose one of the alternatives. In this case,

- Enter **setenv testaction q** to select the quick test.
- Enter **setenv testaction t** to select the thorough test.

You can use the powerup script to run the power-up self-tests without turning the power off and on again. To run the powerup script, at the console prompt (>>) enter

powerup

and press Return.

5.4 Console Mode Tests

From the console prompt (>>), use the t command to run an individual test or the sh command to run a test script. To see a list of available console commands and their formats, at the console prompt (>>), enter

?

and press Return. Appendix A describes the console commands in detail.

5.4.1 Using the t Command

To run an individual test, from the console prompt (>>) enter

t [-l] *slot_number/test_name* [*arg1*] [...] [*argn*]

and press Return.

t	Indicates the test command.
-l	Causes the test to repeat until you press Ctrl-c or reset the system by pushing the Halt button or by switching the power off and then on.
slot number	Replace with the slot number of the module to be tested.
test name	Replace with the name of the test to be run.
arg1...argn	Specify individual test conditions.

Table 5–1: Slot Numbers in Test Commands

Slot Number	Component Tested
0	Option module in slot 0 (on left side, viewing from the back)
1	Option module in slot 1 (middle option slot)
2	Option module in slot 2 (on right side, viewing from the back)
3	Base system hardware, which includes — System module — CPU module — Memory modules (SIMMs and NVRAM) — Base system SCSI controller — Base system Ethernet controller — Serial line controllers

5.4.1.1 To display a list of available tests

To display a list of tests available for a module, from the console prompt (>>) enter

t *slot_number*/?

and press Return. Replace *slot_number* with the number of the slot where the module is installed. A display similiar to this appears on the monitor:

```
cache/data       I or D[D]    address[80050000]
cache/isol       I or D[D]
cache/reload     I or D[D]    address[80050000]
cache/seg        I or D[D]    address[80050000]
fpu
mem              board[0]   thrsld[10]   pattern[55555555]
mem/init
mem/float10      address[A0100000]
mem/select
mfg/done
misc/pstemp
misc/wbpart
rtc/nvr           pattern[55]
rtc/period
rtc/regs
rtc/time
tlb/prb
tlb/reg          pattern[55555555]
```

- The first column lists the names of the tests available for the module in the slot that you specified.
- Entries in the other columns are individual test parameters. The value in brackets next to each parameter is the default value for that parameter.

5.4.2 Common Tests

This section briefly describes the following frequently used tests:

- SCSI controller test
- SCSI send diagnostics test
- Ethernet external loopback test
- Transmit and receive test
- Pins test

Appendix B describes the base system module tests and their parameters and error messages in detail. For information about the TURBOchannel module tests, refer to the *TURBOchannel Maintenance Guide*.

5.4.2.1 SCSI controller (cntl) test

To cntl test tests the operation of a SCSI controller. For example, to run the controller test on the base system SCSI controller, at the console prompt (>>) enter

t 3/scsi/cntl

and press Return. For information about SCSI controller test error messages, see the "SCSI Controller Test" section in Appendix B.

5.4.2.2 SCSI send diagnostics (sdiag) test

The sdiag test runs the self-test for an individual SCSI device. For example, to run the SCSI send diagnostics test on device 0 connected to the base system SCSI controller, at the console prompt (>>) enter

t 3/scsi/sdiag

and press Return. For information about sdiag test parameters and error messages, see the "SCSI Send Diagnostics Test" section in Appendix B.

5.4.2.3 Ethernet external loopback test

The Ethernet external loopback test tests an Ethernet controller and its connections. First install a ThickWire loopback connector on the external connector of the controller to be tested. Then, enter the xternal loopback test command. For example, to test the base system Ethernet controller, at the console prompt (>>) enter

t 3/ni/ext-lb

and press Return. For information about external loopback test error messages, see the "SCSI Controller Test" section in Appendix B.

5.4.2.4 SCC transmit and receive test

The SCC transmit and receive test tests the transmit and receive function of a serial port. First, install a communications adapter with an MMJ loopback connector on the serial connector to be tested, then enter the SCC transmit and receive test command. For example, to run the internal loopback test on serial line 3, at the console prompt (>>) enter

t 3/scc/tx-rx 3 int

and press Return. For information about the SCC transmit and receive test format and error messages, see the "SCC transmit and receive test" section in Appendix B.

5.4.2.5 SCC pins test

The SCC pins test tests the pins on a serial communications connector. First, install a modem loopback connector on the communications connector, then enter the SCC pins test command. For example, to test serial line 3 using a 29-24795 loopback connector, at the console prompt (>>) enter

t 3/scc/pins 3 29-24795

and press Return. For information about the SCC pins test format, the pins tested by the different loopback connectors, and the pins test error messages, see the "SCC Pins Test" section in Appendix B.

5.5 Test Scripts

The ROM for each module contains preprogrammed test scripts. A test script is a short program that includes a list of individual tests and other test scripts. When you run a test script, the system automatically runs the included tests and scripts in order.

Use the sh command to run a test script. To run a test script once and then stop, at the console prompt (>>) enter

sh *slot_number/test_script*

and press Return. Replace *slot_number* with the slot number of the module that you want to test. Replace *test_script* with the name of the test script that you want to run.

For example, to run the quick pst test script on the option module in slot 1, at the console prompt (>>) enter

sh -1/pst-q

and press Return. To have a test script keep repeating until you press Ctrl-c, at the console prompt (>>) enter

sh -l *slot_number/test_script*

and press Return.

For detailed information about scripts, see the "script Command" and "sh Command" sections in Appendix A.

5.5.1 To Display a List of Available Scripts

To display a list of scripts available for a module, from the console prompt (>>) enter

ls *slot_number*

and press Return. Replace *slot_number* with the slot number of the module.

This is a partial listing of the scripts in the base system module:

```
28      1       cnfg -> code
28      1       boot -> code
24      1       rst-q -> rst
24      1       rst-t -> rst
28      1       rst-m -> powerup
32      1       test-ni-m -> test-ni-t
28      1       init -> code
304     1       powerup
44      1       reset
36      1       halt-r
28      1       halt-b
192     1       pst-m
272     1       pst-q
196     1       pst-t
96      1       tech
121     1       test
2401    1       test-cache
132     1       test-cpu
1928    1       test-scc-m
868     1       test-scc-t
124     1       test-crt
60      1       test-misc
268     1       test-mem-m
80      1       test-mem-q
184     1       test-mem-t
196     1       test-ni-t
88      1       test-rtc
40      1       test-scsi
104     1       rst
88      1       cnsltest
```

5.5.2 To Display the Contents of a Script

To see which individual tests and other test scripts are in a specific test script, at the console prompt (>>) enter

cat *slot_number/script_name*

and press Return. Replace *slot_number* with the slot number of the module. Replace *script_name* with the name of the test script for which you want a listing.

The system displays a list of the individual tests and any other test scripts that are in the test script. The following example shows the cat command and the resulting listing of the contents of the test-rtc test script for slot 3 (the base system module):

```
>>cat 3/test-rtc
t ${#}/rtc/regs
t ${#}/rtc/nvr
t ${#}/rtc/period
t ${#}/rtc/time
```

In the listing of a test script, the character # represents the slot number of the module where the script resides.

The cat command displays the contents of test scripts only. It does not display the contents of other objects.

For further information about the cat command, see the "cat Command" section in Appendix A.

5.5.3 To Create a Test Script

You can create a test script to test modules under conditions you choose.

1. At the console prompt (>>), enter

   script *script_name*

   and press Return. Replace *script_name* with the name you want to give the script you are creating.

2. Enter the test commands for the tests that you want to include in the script.

   — Enter test commands in the same order that you want the tests to run. You can include individual tests and test scripts.

   — Specify any test parameters that you want to include with each entry.

   — Press Return after you finish typing each individual test command.

3. To finish creating the test script, press Return twice after you enter the last test command in the test script.

4. To run the script you just created, enter

   sh *script_name*

   and press Return. Replace *script_name* with the name you assigned the script.

The system stores the test script in volatile memory (RAM). The test script is lost when you turn off the system unit or halt the system with the Halt button. You can store only one script at a time.

If you use the ls command to list the test scripts for the base system, the test script you created appears in the test script list.

Appendix A
Console Commands

This appendix explains

- The rules to follow when you type console commands
- Terms commonly used in this discussion of console commands
- The command format and purpose of each console command
- Possible console command error messages

A.1 Using This Appendix

A.1.1 Conventions Used in This Appendix

- **Letters in boldface type like this** are to be typed exactly as they appear.
- *Letters in italic type like this* are variables that you replace with actual values.
- Arguments enclosed in square brackets ([ ]) are optional.
- Ellipses (...) follow an argument that can be repeated.
- A vertical bar (|) separates choices. You can think of the bar as a symbol that means "or".
- Parentheses enclose a group of values from which you must select one value. For example, -(b | h | w) means enter **-b** or **-h** or **-w**.

A.1.2 Some Terms Used in This Appendix

Controller: A hardware device that directs the operation and communication between devices or other controllers. Each controller in the system has a unique controller ID number.

Script: A collection of console commands that run in a set order. Test scripts, which are collections of individual tests and may also contain other test scripts, are commonly used for troubleshooting the system.

Slot: The physical location of a module or modules.

- TURBOchannel option modules occupy slots 0, 1, and 2.
- The base system occupies slot 3. Base system hardware includes the system module, CPU module, and memory modules. The system module contains the base system SCSI and Ethernet controllers.

A.1.3 Rules for Entering Console Commands

You can use console commands when the system monitor displays the >> or R> prompt. When the system displays the R> prompt, you can use only the boot and passwd commands until you enter the console command password.

Follow these rules when you enter console commands:

- Enter uppercase and lowercase letters exactly as they appear in command lines. The system treats uppercase and lowercase letters as different input.
- Press Return after you enter a command.
- Enter numbers as follows:
  - Enter decimal values as a string of decimal digits with no leading zeros (for example, 123).
  - Enter octal values as a string of octal digits with a leading zero (for example, 0177).
  - Enter hexadecimal values as a string of hexadecimal digits preceded by 0x (for example, 0x3ff).
- When reading or writing to memory, enter data as bytes, halfwords, or words. Because a word is 4 bytes, successive addresses referenced by a word are successive multiples of 4. For example, the address following 0x80000004 is 0x80000008. An error occurs if you specify an address that is not on a boundary for the data size you are using.
- When reading or writing to memory, enter the address in kseg1 format. kseg1 format refers to uncached, unmapped address space. In kseg1 format, the uppermost three bits of the address are always 101 and the hexadecimal form of the address always begins with an A or a B. For example, if an address is listed in an error message as 0x04040404, you would use 0xA4040404 to specify that address in a console command. If an address is listed in an error message as 0x14040404, you would use 0xB4040404 to specify that address in a console command.

- The following key combinations have an immediate effect when the system is in console mode:
  - Ctrl-s freezes the screen display.
  - Ctrl-q releases a frozen screen display.
  - Ctrl-c aborts a command.
  - Ctrl-u erases a partially entered line.

A.2 Console Command Reference

This section describes the console commands used to test the following hardware:

- System module
- CPU module
- Memory modules
- Ethernet controllers
- SCSI controllers

Console commands in this appendix appear in the same order as they appear in the system console command Help menu.

For information about console commands used by TURBOchannel options not on this list, refer to the *TURBOchannel Maintenance Guide*.

A.2.1 Console Command Format Summary

Here are the console commands and their formats displayed in the Help menu that appears when you enter **?**:

```
?[cmd]
boot [[-z #] [-n] #/path [ARG...]]
cat SCRPT
cnfg [#]
d [-bhw] [-S #] RNG val
e [-bhwcdoux] [-S #] RNG
erl [-c]
go [ADR]
init [#] [-m] [ARG...]
ls [#]
passwd [-c] [-s]
printenv [EVN]
restart
```

```
script SCRPT
setenv EVN STR
sh [-belvS] [SCRPT] [ARG..]
t [-l] #/STR [ARG..]
unsetenv EVN
```

The following sections describe the console commands in detail. Note that the command descriptions do not always use the format that appears in the Help menu.

Table A–1 lists the console commands.

Table A–1: Console Commands

Command	Function
? [*cmd*]	Displays console commands and formats.
boot [**-z** *seconds*] [**-n**] [*bootpath*] [**-a**] [*args...*]	Boots the system.
cat *slot number /script name*	Displays the contents of a script.
cnfg [*slot number*]	Displays system configuration information.
d [-(**b** | **h** | **w**)] [**-S***count*] *rng*	Deposits data into memory.
e [-(**b** | **h** | **w**)] [**-c**] [**-d**] [**-o**] [**-u**] [**-x**] [**-S***count*] *rng*	Examines memory contents.
erl [**-c**]	Displays the error message log.
go [*address*]	Transfers control to a specific address.
init [*slot number*] [-m]	Resets the system or a module.
ls [*slot number*]	Displays the scripts and other files in a module.
passwd [**-c**] [**-s**]	Sets and clears the console password.
printenv [*variable*]	Prints environment variables.
restart	Attempts to restart the operating system software specified in the restart block.
script *name*	Creates a temporary script of console commands.
setenv *variable value*	Sets an environment variable.
sh [**-b**] [**-e**] [**-l**] [**-v**] [**-S**] [*slot number/script*] [*arg...*]	Runs a script.

Table A–1 (Cont.): Console Commands

Command	Function
t [**-l**] *slot number /test name* [*arg1*]... [*argn*]	Runs a test.
test	Runs a comprehensive test script that checks the system hardware.
unsetenv *variable*	Removes an environment variable.

A.2.2 ? Command

Use the ? command to display a list of available console commands and their formats. The ? command format is

? [*cmd*]

- To display the format for all available console commands, omit the optional *cmd* parameter.
- To display the format for a single command, replace the optional *cmd* parameter with the name of the command for which you want a command format display.

A.2.3 boot Command

Use the boot command to boot the system software. The boot command format is

boot [**-z***seconds*] [**-n**] [*bootpath*] [**-a**] [*args...*]

- Include the optional -z *seconds* parameter to have the system wait before starting the bootstrap operation. Replace *seconds* with the number of seconds the system should wait before the bootstrap operation starts.
- Include the optional -n parameter to have the boot command load, but not execute, the specified file.
- Replace the optional *bootpath* parameter with the specification for the file you are using to boot. The file specification form depends on the type of boot device you use.
  - To boot from Ethernet, use the file specification form

    slot_number/protocol [*/file*]

Replace *slot_number* with the slot number of the Ethernet controller you are using to boot. The protocol parameter represents the name of the network protocol that performs the boot operation.

Replace *protocol* with either mop or tftp. The optional file parameter represents a specific file that you use to boot.

For example, to use the protocol named mop to boot from the base system Ethernet, which uses slot number 3, enter boot 3/mop and press Return.

— To boot from a drive, use the file specification form

slot_number/(**rz** | **tz**) *scsi_id*/*file_name*

Replace *slot_number* with the SCSI controller slot number. Use the (rz | tz) parameter to specify the type of drive that performs the boot operation. Specify rz to boot from a hard disk or compact disc drive. Specify tz to boot from a tape drive.

Replace *scsi_id* with the SCSI ID for the drive you are using to boot.

Replace *file_name* with the name of the specific file you want to boot.

For example, to boot the file named vmunix in multiuser mode from a hard disk drive with SCSI ID 1 that is on the SCSI bus connected to the base system SCSI controller in slot 3, enter

boot 3/rz1/vmunix -a

and press Return.

For example, to boot the file called vmunix from a tape drive that has SCSI ID 2 and is on the SCSI bus connected to a SCSI controller in option slot 1, enter

boot 1/tz2/vmunix

and press Return.

The tape labeled "ULTRIX 4.L supported vol. 1 (RISC)" is the bootable tape.

- To perform a multiuser boot operation, include the -a argument. If you omit the -a argument, the system performs a single-user boot.

A.2.3.1 Important information about the boot command

- If you do not include a boot path in the boot command, the system uses the boot environment variable as the string for the boot command.
- If you include any additional arguments, you must enter the entire string in the boot command. The system ignores the boot environment variable whenever you specify any arguments in the boot command.
- If you use any spaces or tabs in the boot environment variable, you must surround the entire value with double quotation marks. For example, to set the boot environment variable to use the mop protocol to perform a multiuser boot from the base system Ethernet controller in slot 3, enter

  setenv boot "3/mop -a"

  and press Return.
- For details about the boot command parameters for each TURBOchannel option module, refer to the documentation for the TURBOchannel option module in which you are interested.

A.2.4 cat Command

Use the cat command to display the contents of a script. The cat command format is

cat *slot_number/script_name*

- Replace *slot_number* with the slot number of the module that has the contents you want to display.
- Replace *script_name* with the name of the script for which you want to display the contents.

For example, to display the individual self-tests contained in the test-rtc test script in the base system, enter

cat 3/test-rtc

and press Return.

The following list of the individual tests that are in the test-rtc test script then appears on the monitor:

```
>>cat 3/test-rtc
t ${#}/rtc/regs
t ${#}/rtc/nvr
t ${#}/rtc/period
t ${#}/rtc/time
```

A.2.5 cnfg Command

Use the cnfg command to display hardware configuration information. The cnfg command format is

cnfg [*slot_number*]

- To display general system configuration information, enter the cnfg command without the *slot_number* parameter.
- To display detailed configuration information for an individual module, replace the optional *slot_number* parameter with the slot number of the module for which you want a configuration display.

A.2.5.1 General system configuration displays

The following sample general system configuration display is for a system with optional NVRAM, Ethernet, and SCSI modules installed:

```
>>cnfg
3:   KN03-AA    DEC   V5.0a      TCFO      (24MB,  1MB NVRAM)
                                           (enet: 08-00-2b-24-5b-82)
                                           (scsi = 7)

2:   PMAD-AA    DEC   V5.1f      TCFO      (enet: 08-00-2b-0f-43-31)
1:   PMAZ-AA    DEC   V5.1e      TCFO      (scsi = 7)
0:   PMAG-BA    DEC   V5.2a      TCFO      (CX -- D=8)
```

Lines that begin with 0, 1, 2, or 3 describe the modules, if any, that are in the option slots.

- The number that begins the line is the module slot number.
- The second term is the module name.
- The third term is the module vendor.
- The fourth term is the firmware version of the module.
- The fifth term is the type of firmware that is in the module ROM chip.

- The messages in parentheses in the rightmost column provide additional information about each module. The meaning of each message depends on the type of module being described.
  - For the system module, the three lines in this column describe base system hardware. The first line lists the amount of memory in the system. The second line lists the address for the base system Ethernet controller. The third line lists the ID of the base system SCSI controller.
  - For TURBOchannel Ethernet controllers, the additional information is the Ethernet station address.
  - For TURBOchannel SCSI controllers, the additional information is the SCSI ID for the SCSI controller.

Individual configuration displays begin with the same line that describes that module in the general system configuration.

A.2.5.2 Base system configuration displays

To obtain a base system configuration display, enter

cnfg 3

and press Return.

This is a sample configuration display for the base system:

```
3:  KN03-AA   DEC   V5.2A   TCF0  ( 24 MB,     1 MB NVRAM)
                                  (enet: 08-00-2b-0f-45-72)
                                  (SCSI = 7)

          ---------------------------------------------------
          DEV   PID              VID     REV    SCSI DEV
          ====  ==============   ======  =====  ========
          tzl                                   SEQ
          rz2   RZ58    (C) DEC  DEC     nnnn   DIR
          rz4   RRD42   (C) DEC  DEC     nnnn   CD-ROM

     dcache  ( 64 KB), icache ( 64 KB)

     mem(   0):  a0000000:a07fffff  (  8 MB)
     mem(   1):  a0800000:a0ffffff  (  8 MB)
     mem(   2):  a1000000:a17fffff  (  8 MB)
     mem( 14):  a7000000:a70fffff  (  1 MB)   Presto-NVR

     mem( 14):  clean, bat ok, armed
```

Notice that the display begins with the same information as in the general system configuration display. The rest of the display provides details regarding the devices and memory installed in the base slot. This example shows three devices and four memory modules in the base slot.

The configuration display provides the following information about the base system devices:

- The DEV column lists the general category of the drive and its SCSI ID.
  - rz indicates that the drive is a hard disk or optical compact disc drive.
  - tz indicates that the drive is a tape drive.
  - The number at the end of the entry is the drive SCSI ID.
- The PID column lists the product ID for some types of drives.
  - The term on the left indicates the specific drive type.
  - The term on the right indicates the product manufacturer.
- The VID column lists the drive vendor.
- The REV column lists the firmware version number for the drive.
- The SCSI DEV column further describes the drive type.
  - DIR, which indicates a direct access drive, appears in entries for hard disk drives.
  - SEQ, which indicates a sequential access drive, appears in entries for tape drives.
  - CD-ROM appears in entries for optical compact disc drives.

The configuration display provides the following information about the base system memory modules:

- Memory slot number.
- Address range.
- Amount of memory in the slot. The amount can be 8 or 32 megabytes for SIMMs and 1 megabyte for an optional NVRAM module. Except for the NVRAM module, the same amount of memory should be displayed for all the slots because all of the SIMMs should be the same size. The display also will reveal a mixed memory installation, but ULTRIX will not work properly with mixed memory.

A.2.5.3 Ethernet controller configuration displays

To display an Ethernet controller option module configuration, enter

cnfg *slot_number*

Replace *slot_number* with the slot number of the Ethernet controller option module.

To see the base system Ethernet controller configuration, display the base system configuration display. Enter

cnfg 3

and press Return. The base system Ethernet controller configuration is displayed with the other base system configuration information.

The following is a sample Ethernet controller configuration display for an Ethernet controller option module in slot 1:

```
1:  PMAD-AA  DEC   V5.2a   TCF0   (enet: 08-00-2b-0c-e0-d1)
```

The Ethernet controller configuration display has the same meaning as the Ethernet controller description in the general system configuration display. For an explanation of the Ethernet controller configuration display, see the "General System Configuration Displays" section earlier in this appendix.

A.2.5.4 SCSI controller displays

To display a SCSI controller option module configuration, enter

cnfg *slot_number*

and press Return. Replace *slot_number* with the slot number of the SCSI controller option module.

To see the base system SCSI controller configuration, enter

cnfg 3

and press Return. The base system SCSI controller configuration is displayed with the other base system configuration information.

The following is a sample configuration display for a SCSI controller in slot 2 that supports two hard disk drives, one optical compact disc drive, and one tape drive:

```
2:  PMAZ-AA   DEC   V5.2a   TCFO   (SCSI = 7)
           --------------------------------------------
           DEV   PID                VID     REV    SCSI DEV
           ====  =============     ======  =====  ========
           rz0    RZ58    (C) DEC   DEC     nnnn    DIR
           rz1    RZ57    (C) DEC   DEC     nnnn    DIR
           rz4    RRD42   (c) DEC   DEC     nnnn    CD-ROM
           tz5                                      SEQ
```

In the SCSI configuration display, the first line has the same meaning as the SCSI description in the general system configuration display. For an explanation of this first line, see the section "General System Configuration Displays" earlier in this appendix.

Lines following the first line describe drives on the SCSI bus.

- The DEV column lists the general category of the drive and its SCSI ID.
  - rz indicates that the drive is a hard disk or optical compact disc drive.
  - tz indicates that the drive is a tape drive.
  - The number at the end of the entry is the drive SCSI ID.
- The PID column lists the product ID for some types of drives.
  - The term on the left indicates the specific drive type.
  - The term on the right indicates the product manufacturer.
- The VID column lists the drive vendor.
- The REV column lists the firmware version number for the drive.
- The SCSI DEV column further describes the drive type.
  - DIR, which indicates a direct access drive, appears in entries for hard disk drives.
  - SEQ, which indicates a sequential access drive, appears in entries for tape drives.
  - CD-ROM appears in entries for optical compact disc drives.

A.2.6 d Command

Use the d command to deposit values in memory.

The d command format is

d [(-(**b** | **h** | **w**)] [-S*count*] *rng*

- Use one of the optional parameters -(b | h | w) to specify whether to deposit the contents as bytes, halfwords, or words.
  - Specify -b to deposit the contents as bytes.
  - Specify -h to deposit the contents as halfwords.
  - Specify -w to deposit the contents as words.
- Include the optional -S*count* parameter to store the same value more than once. Replace *count* with the number of times that you want the value to be stored.
- Use the *rng* parameter to set the range of addresses across which the values are stored.
  - To deposit values at a single address, replace *rng* with that address.
  - To deposit a number of values across a range of addresses, replace *rng* with the address range. Use the form

    address_low:address_high

    to define the range. Replace emphasis>(address_low) with the starting address for storing values and replace *address_high* with the ending address for storing values.
  - To deposit values at a series of addresses, replace *rng* with the starting address and the number of successive addresses at which you want to store values. Use the form

    address_low#count

    to specify the addresses where you store values. Replace *address_low* with the starting address for storing values. Replace *count* with the number of values you want to store.
  - To specify more than one address range, separate the range specifications with commas. Leave no spaces between the range specifications.

A.2.7 e Command

Use the e command to examine the contents of a specific address. The e command format is

e [**-b** | **h** | **w**] [**-c**] [**-d**] [**-o**] [**-u**] [**-x**] [**-S** *count*] *rng*

- Use one of the optional parameters (-b | h | w) to specify whether to examine the contents as bytes, halfwords, or words.
  - Specify -b to examine the contents as bytes.

- Specify -h to examine the contents as halfwords.
- Specify -w to examine the contents as words.

- Specify -x to display the contents in hexadecimal format.
- Specify -o to display the contents in octal format.
- Specify -u to display the contents in unsigned decimal format.
- Specify -d to display the data in decimal format.
- Specify -c to display the data as ASCII characters.
- Include the optional -S*count* parameter to have the command repeatedly fetch the value, but display the value only once. When you enter this parameter, replace *count* with the number of times that you want to fetch the value.
- Use the *rng* parameter to specify the range of addresses you want to examine.
  - To examine values at a single address, replace *rng* with that address.
  - To examine values at a range of addresses, replace *rng* with the address range. Use the form

    address_low:address_high

    to define the range. Replace *address_low* with the starting address for storing values and replace *address_high* with the ending address for storing values.
  - To examine values at a series of addresses, replace *rng* with the starting address and the number of successive addresses you want to examine. Use the form

    address_low#count

    to specify the addresses where you store values. Replace *address_low* with the starting address for storing values. Replace *count* with the number of addresses at which you want to store values.
  - To specify more than one address range, separate each range specification with commas. Leave no spaces between the ranges.

A.2.8 erl Command

Use the erl command to display or clear the log of the errors that occurred since the most recent power-up or reset operation. When the buffer that holds these error log fills up, no further errors are recorded. If you intend to run tests and use these logs for information, use the erl -c command to clear the logs first. The erl command format is

erl [-c]

- To display the current error message log, use the erl command without the -c option.
- To clear the error message log, include the -c option. When the error log buffer is full, no more messages are added until the buffer is cleared by the erl -c command.

A.2.9 go Command

Use the go command to transfer system control to a specific system address. The go command format is

go [*address*]

- To transfer system control to the address specified in the last boot -n command, the go command without the address parameter. If you omit the address parameter, and if no previous boot -n command has been issued, the system ignores the go command.
- To transfer system control to the contents of a specific address, include the *address* parameter. Replace *address* with the address to which you want to transfer control.

A.2.10 init Command

Use the init command to initialize module hardware. The init command format is

init [*slot_number*] [**- m**]

- To initialize the entire system, specify the init command with no additional arguments.
- To initialize an individual module, replace the optional *slot_number* parameter with the slot number of the module that you want to initialize.
- If you perform an init operation on the system module (slot 3), include the optional -m parameter to zero all memory modules in the system module.

A.2.11 ls Command

Use the ls command to list the scripts and other objects that are in system ROM. The ls command format is

ls [*slot_number*]

To display a list of scripts or other objects that are available in an individual module, replace the optional *slot_number* parameter with the slot number of the module that contains the files you want to display.

This sample display is a portion of the ls display for the base system in slot 3:

```
>>ls 3
     28 1 cnfg -> code
     28 1 boot -> code
     24 1 rst-q -> rst
     24 1 rst-t -> rst
     28 1 rst-m ->powerup
     32 1 test-ni-m -> test-ni-t
     28 1 init -> code
    304 1 powerup
     44 1 reset
     36 1 halt-r
     28 1 halt-b
    192 1 pst-m
    272 1 pst-q
    196 1 pst-t
     96 1 tech
    156 1 test
```

The third column lists the names of the scripts and other objects in the module ROM in the specified slot.

A.2.12 passwd Command

Use the passwd command to enter, set, or clear a password. The passwd command format is

passwd [**-c**] [**-s**]

If the console prompt is R>, you can use only the boot and passwd commands until you enter the correct password.

To enter an existing password, enter the passwd command without any additional parameters. At the pwd: prompt, enter the password. Then press Return.

After you enter the correct password, or if the system does not require a password, the system displays the console prompt >>. You can use all console commands whenever the console prompt is >>.

- To clear an existing password, include the -c parameter when you enter the passwd command. First use the passwd command to enter the existing password. After the console prompt >> appears, enter

  passwd -c

  and press Return. The system then removes the password requirement.

- To set a new password, include the -s parameter. Enter the new password at the pwd: prompt. When the pwd: prompt appears a second time, enter the password again. If the two password entries match, the system set the new value as the password.

- The password must have at least six and no more than 32 characters. The system treats upper-case and lower-case letters as different characters.

A.2.13 printenv Command

Use the printenv command to display the list of environment variables. The printenv command format is

printenv [*variable*]

- To display the entire environment variable table, omit the optional *variable* parameter.
- To display an individual environment variable, replace *variable* with the name of the environment variable you want to display.

A.2.14 restart Command

Use the restart command to restart the system software. For the restart operation to succeed, the operating system software must have a restart block set up in memory. The restart command format is **restart**

A.2.15 script Command

Use the script command to create a temporary set of console commands that run in an order that you specify. The script command format is

script *name*

Replace *name* with the name that you are giving the script.

After you press Return, enter the commands that you want to include in the script. Press Return after each command that you enter. Commands can be t or sh commands. Enter one command per line. When you finish typing the commands that you are including, press Ctrl-d or press Return twice to complete the script.

To run the script, use the sh command described later in this appendix. When you run the script, the commands execute in the order in which you entered them when you created the script.

A.2.16 setenv Command

Use the setenv command to change an environment variable. Table A–2 lists the standard environment variables. When you change a standard environment variable (except osconsole and #), the system stores the new value in NVR and uses it until you use the setenv command to change it again or reset the NVR with the clear-NVR jumper. The setenv command format is

setenv *variable value*

When you enter the setenv command,

- Replace *variable* with the name of the environment variable you want to set.
- Replace *value* with the new value that you want to assign to the environment variable. Note that if the new value contains blank spaces or tabs, you must use double quotation marks (") at the beginning and end of the value.

Table A–2: Environment Variables in the Environment Variable Display

Environment Variable	Description
boot	Sets the default boot path. See the "boot Command" and "unsetenv Command" sections in Appendix C.
console	Selects the system console. — Set the console variable to s to enable the terminal connected to the left comm connector (viewed from the back) as the active console.

Table A–2 (Cont.): Environment Variables in the Environment Variable Display

Environment Variable	Description
EPAWS	Specifies the way the system responds when a diagnostic test finds an error. — Set the EPAWS variable to EPAWS to cause the system to pause when a diagnostic test finds an error. Press any key to continue testing. — If the EPAWS variable is set to any value other than EPAWS, the system does not pause when an error occurs.
haltaction	Specifies the way the system responds when it halts. — Set the haltaction variable to b to cause the console to boot after the console performs the appropriate initialization and self-tests. — Set the haltaction variable to h to cause the console to halt and attempt no other action. — Set the haltaction variable to r to cause the console to restart and then attempt to boot if the restart operation fails.
more	Sets the way the screen scrolls lines of text. — Set the more variable to 0 to have text scroll to the end before stopping. — Set the more variable to a number other than zero to have scrolling pause after that number of lines has been displayed.
osconsole	Contains the slot numbers of the console drivers. If a TTY driver from slot x serves as the console, osconsole is set to x. If a CRT driver from slot y and a kbd driver from slot z serve as the console, osconsole is set to y,z. Although the environment variable display includes the osconsole setting, you cannot set this variable. The system automatically sets the osconsole value.
testaction	Specifies the type of power-up self-test that the system runs. — Specify q to run a quick test when the power-up self-test runs. — Specify t to specify a thorough test when the power-up self- test runs.
#	Specifies the slot number of the module that contains the current script. If no script is active, the system specifies the base system module, slot number 3. Although the environment variable display includes the # setting, you cannot set this variable.

A.2.17 sh Command

Use the sh command to run a script. The sh command format is

sh [-b] [-e] [-l] [-v] [-S] [*slot_number/script*] [arg...]

- Include the optional -b parameter to execute the script directly, instead of through a subshell.
- Include the optional -e parameter to stop the script if an error occurs.
- Include the optional -l parameter to have the script loop until you press Ctrl-c.
- Include the optional -v parameter to echo the script to the console as the test runs.
- Include the optional -S parameter to suppress any error messages if the script is not found.
- To test a specific module with a specific script, include the optional *slot_number/script* parameter.
  - Replace *slot_number* with the slot number of the module that has the script you want to run.
  - Replace *script* with the name of the script you want to run.
- The use of any additional arguments depends on the particular script you specified in the sh command.

You can also run a script by typing *slot_number/script*.

- Replace *slot_number* with the slot number of the module that contains the script you want to run.
- Replace *script* with the name of the script you want to run.

For example, to run the thorough power-up self test script for a SCSI controller and drives in slot 2, enter

sh 2/pst-t

and press Return.

A.2.18 t Command

Use the t command to run individual tests. The t command format is

t [-l] *slot_number/test_name* [*arg...*]

- Include the optional -l parameter to have the test loop until you press Ctrl-c or reset the system.

- Replace *slot_number* with the slot number of the module that you want to test.
- Replace *test_name* with the name of the individual test you want to run.
- The uses of any additional arguments depend on the particular test you are running. For an explanation of the additional arguments used in an individual test, see Appendix B.

To display the name and format of all individual tests for a module, enter

t *slot_number*/?

and press Return. Replace *slot_number* with the slot number of the module for which you want to display tests.

A.2.19 test Command

Use the test command to run a thorough test of all system hardware. The test command format is

test

A.2.20 unsetenv Command

Use the unsetenv command to remove an environment variable. The unsetenv command removes a standard environment variable (except osconsole and #) during the current session only. When the system is reset, reinitialized, or powered up, the values of the standard environment variables revert to their previously set values. Table A–2 section in this appendix lists the standard environment values.

The unsetenv command format is

unsetenv *variable*

When you enter the unsetenv command, replace *variable* with the name of the environment variable that you want to remove.

NOTE: *To clear the boot environment variable, use the setenv command. Enter*

setenv boot ""

and press Return.

A.3 Console Command Error Messages

Table A–3 lists the error messages that the console commands can return.

Table A–3: Console Command Error Messages

Error Message	Meaning
?EV:*ev name*	The specified environment variable does not exist.
?EVV:*value*	The specified environment variable value is invalid.
?IO:*slot_number/device*	An I/O device reported an error; *slot_number* represents the I/O device slot number, and emphasis>(device) represents an additional message about the error.
?IO:*slot_number/device*	The module in the slot represented by *slot_number* does not recognize the device represented by *device*.
?PDE3: *slot_number*	The module in the slot represented by *slot_number* contains an early version of the firmware. The ROM chip must be upgraded.
?SNF: *script*	The system did not find the script that was to be run.
?TXT:	The name specified in the script command is not a valid script name.
?STF (4: Ln#0 Pntr self test)	A pointing-device self-test failed. This is an information message and does not prevent the system from automatically booting.
?STX: *usage*	A console command contained a syntax error. The *usage* parameter lists the correct syntax.
?STX: *error*	A console command contained a syntax error. The error parameter lists the incorrect portion of the command.
?TFL:*slot_number/test*	A test failed; *slot_number* represents the slot number of the module that reported the error. *test* represents the name of the failed test.
?TNF:	The specified test could not be found. The test name was probably entered incorrectly.

Appendix B

Base System Self-Test Commands and Error Messages

This appendix describes commands and messages for the following base system tests:

- System module tests
- CPU module tests
- Memory module tests
- Base system SCSI controller tests
- Base system Ethernet controller tests
- Initial power-up tests

B.1 Locating Individual Tests in This Appendix

When an individual test fails, the name of the test appears in the error message. For details of each base system test, see the section in this appendix that describes the test and its error messages. The tests are listed in alphabetical order.

When troubleshooting the system, you can use the test command to run any single test when the console prompt (>>) appears. You can also write a test script to run a group of individual tests. See the "t Command" and "script Command" sections in Appendix A for more information.

To help you select the individual tests that apply to a problem that you are troubleshooting, Table B–1 lists the individual base system module tests grouped by the function that they test.

Table B–1: Base System Module Tests and Utilities

Test or Utility	Command
	Base System Module Tests
Halt button test	**t 3/misc/halt** [*number*]
Overheat detect test	**t 3/misc/pstemp**
Nonvolatile RAM (NVR) test	**t 3/rtc/nvr** [*pattern*]
Real-time clock period test	**t 3/rtc/period**
Real-time clock registers test	**t 3/rtc/regs**
Real-time test	**t 3/rtc/time**
	Serial Communication Tests
SCC DMA test	**t 3/scc/dma** [*line*] [*loopback*] [*baud*]
SCC interrupts test	**t 3/scc/int** [*line*]
SCC input/output (I/O) test	**t 3/scc/io** [*line*] [*loopback*]
SCC pins test	**t 3/scc/pins** [*line*] [*attachment*]
SCC transmit and receive test	**t 3/scc/tx-rx** [*line*] [*loopback*] [*baud*] [*parity*] [*bits*]
	CPU Module Tests
Cache data test	**t 3/cache/data** [*cache*] [*address*]
Cache fill test	**t 3/cache/fill** [*cache*] [*offset*]
Cache isolate test	**t 3/cache/isol** [*cache*]
Cache reload test	**t 3/cache/reload** [*cache*] [*offset*]
Cache segment test	**t 3/cache/seg** [*cache*] [*address*]
CPU-type utility	**t 3/misc/cpu-type**
Floating-point unit (FPU) test	**t 3/fpu**
Translation lookaside buffer (TLB) probe test	**t 3/tlb/prb**
TLB registers test	**t 3/tlb/reg** [*pattern*]
	Memory Module Tests

Table B–1 (Cont.): Base System Module Tests and Utilities

Test or Utility	Command
Error correction coding (ECC) correction test	**t 3/ecc/cor** [*address*]
Memory module test	**t 3/mem** [*module*] [*threshold*] [*pattern*]
Floating I/O memory test	**t 3/mem/float10** [*address*]
Zero memory utility	**t 3/mem/init**
RAM select lines test	**t 3/mem/select**
Partial write test	**t 3/misc/wbpart**
	NVRAM Prcache Tests
Prcache quick test	**t 3/prcache**
Disable NVRAM battery	**t 3/prcache arm**
Enable NVRAM battery	**t 3/prcache unarm**
	Ethernet Controller Tests
Collision test	**t 3/ni/cllsn**
Cyclic redundancy code (CRC) test	**t 3/ni/crc**
Display maintenance operation protocol (MOP) counters utility	**t 3/ni/ctrs**
Ethernet-direct memory access (DMA) registers test	**t 3/ni/dma1**
Ethernet-DMA transfer test	**t 3/ni/dma2**
Ethernet station address ROM (ESAR) test	**t 3/ni/esar**
External loopback test	**t 3/ni/ext-lb**
Interrupt request (IRQ) test	**t 3/ni/int**
Internal loopback test	**t 3/ni/int-lb**
Multicast test	**t 3/ni/m-cst**
Promiscuous mode test	**t 3/ni/promisc**
Registers test	**t 3/ni/regs**

Table B–1 (Cont.): Base System Module Tests and Utilities

Test or Utility	Command
	SCSI Tests
SCSI controller test	**t 3/scsi/cntl**
SCSI send diagnostics test	**t 3/scsi/sdiag** *scsi_id* [**d**] [**u**] [**s**]
SCSI target test	**t 3/scsi/target** *scsi_id* [**w**] [**l***loops*]

B.2 Tests

The following sections explain the commands, parameters, and error messages for each base system module test. The tests are presented in alphabetical order.

B.2.1 cache/data - Cache Data Test

The cache data test writes data patterns to the cache and then reads them. To run the cache data test, enter

t 3/cache/data [*cache*] [*address*]

and press Return. When you enter the cache test command,

- Replace *cache* with a value that specifies the cache you want to test.
  - Specify I to test the instruction cache.
  - Specify D to test the data cache. The default value is D.
- Replace the optional *address* parameter with the specific cache address where you want the test to start. Using the address parameter requires familiarity with the firmware specifications. The default address is 80500000.

B.2.1.1 Cache data test error messages

Cache data test error messages have the form

```
?TFL:3 /cache/data (code:
[address=actual, sb expected])
```

- `?TFL 3/cache/data` indicates that the cache data test reported an error.

- `code` represents a number that identifies the portion of the test that failed.
- The optional `address=actual, sb expected` phrase indicates the expected and actual values in the cache.
  - `address` represents the address where the error occurred.
  - `actual` represents the actual value at that address.
  - `expected` represents the expected value for that address.

Table B–2 lists the codes used in cache data test error messages.

Table B–2: Cache Data Test Error Codes

Error Code	Description
1	Error occurred writing data pattern to cache RAM.
2	Cache parity error occurred while test was reading floating 1.
3	Error occurred when test read data pattern in cache.
4	Cache parity error occurred while test was reading floating 0.
5	Error occurred when test wrote address complement to cache RAM.
6	Cache parity data error occurred.
7	Error occurred reading address complement.
8	Cache address read caused a parity error.

B.2.2 cache/fill - Cache Fill Test

The cache fill test writes rotating data patterns to memory in spans that are twice the size of the cache and then reads the patterns. To run the cache fill test, enter

t 3/cache/fill [*cache*] [*offset*]

and press Return. When you enter the cache fill test command,

- Replace *cache* with a value that specifies the cache you want to test.
  - Specify I to test the instruction cache.
  - Specify D to test the data cache. The default value is D.
- Replace *offset* with a specific cache address where you want the test to start. The default address is 80500000.

B.2.2.1 Cache fill test error messages

Cache fill test error messages have the form

```
?TFL: 3/cache/fill (description)
```

- `?TFL 3/cache/fill` indicates that the cache fill test reported an error.
- *description* represents an additional message that describes the error.

Table B–3 lists the descriptions used in cache fill test error messages.

Table B–3: Cache Fill Test Error Descriptions

Error Description	Meaning
(PE)	Unexpected parity error occurred.
(address= actual, sb expected)	Data pattern read reported a miscompare. *address* represents the address where the miscompare occurred. *actual* represents the actual value at that address. *expected* represents the expected value for that address.
(PE @ address (C))	Parity error occurred. The address parameter lists the address where the error occurred.

B.2.3 cache/isol - Cache Isolate Test

The cache isolate test isolates data patterns to the cache and then reads and compares them. To run the cache isolate test, enter

t 3/cache/isol [*cache*]

and press Return. When you enter the cache isolate test command, replace *cache* with a value that specifies the cache you want to test.

- Specify I to test the instruction cache.
- Specify D to test the data cache. The default value is D.

B.2.3.1 Cache isolate test error messages

Cache isolate test error messages have the form

```
?TFL: 3/cache/isol (code:
[address=actual,
sb expected])
```

- ?TFL 3/cache/isol indicates that the cache isolate test reported an error.
- *code* represents a number that identifies the portion of the test that failed.
- The optional phrase *address= actual, sb expected* indicates the actual and expected values at the address where the error occurred.
  - *address* represents the address where the error occurred.
  - *actual* represents the actual value at that address.
  - *expected* represents the expected value at that address.

Table B–4 lists the codes used in cache isolate test error messages.

Table B–4: Cache Isolate Test Error Codes

Error Code	Description
1	Reading 00000000 pattern resulted in a cache parity error.
2	Reading 00000000 pattern resulted in a cache miss error.
3	Reading 00000000 pattern returned a data miscompare.
4	Reading 55555555 pattern resulted in a cache parity error.
5	Reading 55555555 pattern resulted a cache miss error.
6	Reading 55555555 pattern resulted in a data miscompare.
7	Reading AAAAAAAA pattern resulted in a cache parity error.
8	Reading AAAAAAAA pattern resulted in a cache miss error.
9	Reaaing AAAAAAAA pattern resulted in a data miscompare.
10	Reading data address pattern resulted in a cache parity error.
11	Reading data address pattern resulted in a parity error.
12	Reading data address pattern returned a miscompare error.

B.2.4 cache/reload - Cache Reload Test

The cache reload test writes rotating-parity data patterns to memory and then reads the patterns. To run the cache reload test, enter

t 3/cache/reload [*cache*] [*offset*]

and press Return. When you enter the cache reload test command,

- Replace *cache* with a value that specifies the cache you want to test.
  - Specify I to test the instruction cache.
  - Specify D to test the data cache. The default value is D.
- Replace *offset* with a specific cache address where you want the test to start. The default address is 80500000.

B.2.4.1 Cache reload test error messages

Cache reload test error messages have the form

```
?TFL: 3/cache/reload (description)
```

- `?TFL 3/cache/reload` indicates that the cache reload test reported an error.
- *`description`* represents an additional message that describes the error.

Table B–5 lists the descriptions used in cache reload test error messages.

Table B–5: Cache Reload Test Error Descriptions

Error Description	Meaning
(PE)	Unexpected parity error occurred.
(*address= actual*, sb *expected*)	Data pattern read reported a miscompare. *address* represents the address where the miscompare occurred. *actual* represents the actual value at that address. *expected* represents the expected value for that address.
(PE @ address (C))	Parity error occurred. The address parameter lists the address where the error occurred.

B.2.5 cache/seg - Cache Segment Test

The cache segment test checks individual cache segments. To run the cache segment test, enter

t 3/cache/seg [*cache*] [*address*]

and press Return. When you enter the cache segment test command,

- Replace the optional *cache* parameter with a value that specifies the cache you want to test.
  - Specify D to test the data cache. The default value is D.
  - Specify I to test the instruction cache.
- Replace *address* with a specific address you want to test. The default address is 80500000. Note that using the optional address parameter correctly requires thorough knowledge of the firmware specifications.

B.2.5.1 Cache segment test error messages

Cache segment test error messages have the form

```
?TFL: 3/cache/seg (code : description)
```

- `?TFL 3/cache/seg` indicates that the cache segment test reported an error.
- *`code`* represents a number that identifies the portion of the test that failed.
- *`<description`* represents additional information that describes the failure.

Table B–6 lists the codes and descriptions used in cache segment test error messages.

Table B–6: Cache Segment Test Error Codes and Descriptions

Error Code and Description	Meaning
(1: address=*xxxxxxxx*, sb *yyyyyyyy*)	Error occurred when the system tried to read the cache contents. The address parameter is the actual value at a given address. The correct value follows.
(2: address=*xxxxxxxx*, sb *yyyyyyyy*)	Error occurred when the system tried to read the memory contents. The address parameter is the actual value at a given address. The correct value follows.
(3: address=*xxxxxxxx*, sb *yyyyyyyy*)	Error occurred when the system performed a read and write operation on the uncached memory. The address value is the actual value at a given address. The correct value follows.
(4: address =*xxxxxxxx*, sb *yyyyyyyy*	Cache data was inconsistent. The address value is the actual value at a given address. The correct value follows.

B.2.6 ecc/cor - Error Correction Coding (ECC) Correction Test

The error correction coding (ECC) correction test writes data patterns XOR'd with floating ones to create single bit errors and then checks to see whether the error was detected and corrected. To run the ECC correction test, enter

t 3/ecc/cor [*address*]

and press Return. Replace *address* with a specific address you want to test. The default address is A0010000.

B.2.6.1 ECC correction test error messages

ECC correction test error messages have the form

```
?TFL: 3/ecc/cor (code:description)
```

- `?TFL 3/ecc/cor` indicates that the ECC correction test reported an error.
- *`code`* represents a number that identifies the portion of the test that failed.
- *`description`* represents additional information that describes the failure.

Table B–7 lists the codes and descriptions used in ECC correction test error messages.

Table B–7: ECC Correction Test Error Codes and Descriptions

Error Code and Description	Meaning
(1: *xxxxxxxx* rd err) [KN03-AA]	Cannot read and write location with good data.
(2: sbe not det)	Single bit error in the data is not detected.
(3: sbe not cor)	Single bit error in the data is not corrected.

B.2.7 fpu - Floating-Point Unit Test

The floating-point unit (FPU) test uses the FPU to perform simple arithmetic and compares the result to known values. To run the FPU test, enter

t 3/fpu

and press Return.

B.2.7.1 FPU test error messages

FPU test error messages have the form

```
?TFL:  3/fpu  (code)
```

- `?TFL 3/fpu` indicates that the FPU test reported an error.
- *code* represents a number that identifies the portion of the test that failed.

Table B–8 lists the codes used in FPU test error messages.

Table B–8: FPU Test Error Codes

Error Code	Meaning
1	Values did not match. Value should be 00000000.
2	Values did not match. Value should be 55555555.
3	Values did not match. Value should be AAAAAAAA.
4	Values did not match. Value should be FFFFFFFF.
5	Least-significant bit failed when the system was converting doubleword to word (CVT D. W.)
6	Most-significant bit failed when the system was converting doubleword to word (CVT D. W.)
7	Double miscompare occurred: n+n-n!=n
8	Double miscompare occurred: n+n= =n
9	Convert float-double. Value should be 55555555.
10	FPU CSR double error occurred.
11	Single miscompare occurred: n+n-n!=n
12	Single miscompare occurred: n+n= =n
13	Convert float-double. Value should be 55555555.
14	FPU CSR single error occurred. Value should be 00000000.
15	Single division failed. Value should be 00005555.
16	Single multiplication failed.
17	Double multiplication failed.
18	Double division failed.
19	Conversion error occurred. Pattern readback did not match.
21	FPU did not trap on overflow exception.
22	Did not get FPU interrupt.

B.2.8 mem - Memory Module Test

The memory module test performs a full pattern test on an entire SIMM or NVRAM memory module. To run the memory module test, enter

t 3/mem [*module*] [*threshold*] [*pattern*] and press Return. When you enter the memory module test command,

- Include the *module* parameter to specify the memory module you want to test.
  - To test one memory module, specify the slot number of the memory module that you want to test. The default value is 0.
  - To test all memory modules, specify an asterisk (*).
- Replace the optional *threshold* parameter with the number of single-bit errors the test allows before the test fails. The default threshold is 10.
- Replace the optional *pattern* parameter with a specific pattern that you want to use in the test. The default pattern is 55555555.

B.2.8.1 Memory module test error messages

Set the verbose environment variable to 1 to see compare error messages in the following form:

`?TFL:3/mem @ address=actual, expected`

- `?TFL 3/mem @` indicates that the memory test reported a compare error.
- The `address` parameter is the address at which the error occurred.
- `actual` represents the value at that address.
- `expected` represents the expected value at that address.

If the verbose environment variable is not set, the error messages appear in the following formats:

`?TFL:3/mem (1: board L, MBE=M, SBE=N)`

`?TFL:3/mem (2: board L, too many SBEs:N)`

where `L` represents the slot number of the failed memory module, `M` represents the number of multibit errors that occurred, and `N` represents the number of single-bit errors that occurred.

B.2.9 mem/float10 - Floating 1/0 Memory Test

The floating 1/0 memory test writes floating 1 and floating 0 across one location in RAM or NVRAM. To run the floating 1/0 memory test, enter

t 3/mem/float10 [*address*]

and press Return. When you enter the floating 1/0 memory test command, replace the optional *address* parameter with a specific address at which you want to start writing 1s. The default address is A0100000.

B.2.9.1 Floating 1/0 memory test error messages

If a RAM module is tested, the floating 1/0 memory test error message is

```
?TFL: 3/mem/float10 (Err= N)
```

where *N* represents the number of errors the memory module reported.

If an NVRAM module is tested and the module contains valid data, the floating 1/0 memory test error message is

```
?TFL: (1: (tst nocomp)
```

B.2.10 mem/init - Zero Memory Utility

The zero memory utility floods RAM and NVRAM modules with zeros as fast as possible. If the NVRAM module contains valid data, only the scratch area will be cleared. To run the zero memory utility, enter

t 3/mem/init

and press Return.

The zero memory utility returns no error codes.

B.2.11 mem/select - RAM Select Lines Test

The RAM select lines test checks for RAM select line faults by performing a read and write operation on one location in each memory module. To run the RAM select lines test, enter

t 3/mem/select

and press Return.

B.2.11.1 RAM select lines test error messages

The only RAM select test error message is

```
?TFL: 3/mem/select (address=actual, sb expected)
```

- `?TFL: 3/mem/select` indicates that the RAM select lines test reported an error.
- *address* represents the memory address where the error occurred.
- *actual* represents the actual value at the listed address.

- *expected* represents what the value at the listed address should be.

B.2.12 misc/cpu-type - CPU-Type Utility

The CPU-type utility displays a message that identifies the CPU type. To run the CPU-type utility, enter

t 3/misc/cpu-type

and press Return.

B.2.12.1 CPU-type utility messages

The CPU-type utility message has the form

```
3/misc/cpu-type's code: NDX-type
```

where `type` represents a code that indicates the type of CPU module installed in the system. For example, the code `NDX-129A` identifies the KN03-GA CPU module (40-MHz).

B.2.13 misc/halt - Halt Button Test

The halt button test checks whether the halt button is connected and can generate an interrupt. To run the halt button test, enter

t 3/misc/halt [*number*]

and press Return.

When you enter the halt button command, replace *number* with the number that specifies the type of test you want to run.

- Specify 0 to check whether the halt button is pressed. If you specify 0 and the button is pressed when the test runs, the test reports an error. The default value is 0.
- Specify a number from 1 to 9 to check whether the button responds when pressed. Press the button the same number of times as the number you specify in the test command.

B.2.13.1 Halt button test error messages

There are two halt button test error messages.

- `?TFL: 3/misc/halt (1:SIR=xxxxxxxx)`

  This message indicates that the halt button is pressed in. `xxxxxxxx` represents the value in the system interrupt register.

- `?TFL: 3/misc/halt (2: invld bits: SIR=xxxxxxxx)`

This message indicates that the system interrupt register contains an impossible combination of halt-button bits. *xxxxxxxx* represents the value in the system interrupt register.

B.2.14 misc/pstemp - Overheat Detect Test

The overheat detect test checks whether the power supply is overheating. To run the overheat detect test, enter

t 3/misc/pstemp

and press Return.

B.2.14.1 Overheat detect test error message

When the overheat detect test fails, the following error message is displayed:

```
?TFL: 3/misc/pstemp (system is *HOT*)
```

This message indicates that the system is overheating.

B.2.15 misc/wbpart - Partial Write Test

The partial write test writes to a specific memory address and then checks whether the written values are correct. To run the partial write test, enter

t 3/misc/wbpart

and press Return.

B.2.15.1 Partial write test error messages

Partial write test error messages have the form

```
?TFL: 3/misc/wbpart (code)
```

- `?TFL 3/misc/wbpart` indicates that the partial write test reported an error.
- *code* represents a number that identifies the portion of the test that failed.

Table B–9 lists the codes used in partial write test error messages.

Table B–9: Partial Write Test Error Codes

Error Code	Meaning
1	Pattern that was read showed mismatch on word access.
2	Byte 0 failed partial byte write.
3	Byte 1 failed partial byte write.
4	Byte 2 failed partial byte write.
5	Byte 3 failed partial byte write.
6	Halfword 0 failed partial halfword write.
7	Halfword 1 failed partial halfword write.

B.2.16 ni/cllsn - Collision Test

The collision test checks Ethernet collision detect circuitry by forcing a collision on transmission. To run the collision test, enter

t 3/ni/cllsn

and press Return.

B.2.16.1 Collision test error messages

Collision test error messages have the form

```
?TFL: 3/ni/cllsn (code:description)
```

- `?TFL: 3/ni/cllsn` indicates that the collision test reported a problem.
- *code* represents a number that identifies the portion of the test that failed.
- *description* represents additional information that describes the failure.

Table B–10 lists the codes and descriptions used in collision test error messages.

Table B–10: Collision Test Error Codes and Descriptions

Error Code and Description	Meaning
3: cllsn not dtctd	Ethernet controller chip failed to detect an Ethernet collision.
4: xmt [x]	Ethernet controller chip transmission failed.
6: LANCE-init [x]	Ethernet controller chip failed to initialize.

B.2.17 ni/common - Common Diagnostic Utilities

The common diagnostic utilities are run by Ethernet controller tests. You cannot run these diagnostic utilities by themselves.

B.2.17.1 Common diagnostic utility error messages

Common diagnostic utility error messages have the form

```
?TFL: 3/ni/test_name (code:description)
```

- `?TFL: 3/ni` indicates that a common diagnostic utility detected an error.
- *test_name* represents the name of the test in which the diagnostic utility detected an error.
- *code* represents a number that identifies the utility that generated the error message.
- *description* represents additional information that describes the error.

Table B–11 lists the codes and descriptions used in common diagnostic utility error messages.

Table B–11: Common Diagnostic Utility Error Codes and Descriptions

Error Code: Description	Meaning
700: Invld param frmt [*xxxxx*]	Parameter *xxxxx* was not in a valid format.
901: err hltng LANCE	STOP bit did not halt the Ethernet controller chip.
902: LANCE-init timeout	Timeout occurred when the system tried to initialize the Ethernet controller chip.

Table B–11 (Cont.): Common Diagnostic Utility Error Codes and Descriptions

Error Code: Description	Meaning
903: LANCE-start timeout	Timeout occurred waiting for the Ethernet controller chip to start.
904: err initing LANCE	Utility could not initialize the Ethernet controller chip.
905: LANCE-stop timeout	Timeout occurred in the Ethernet controller chip.
906: err initing LANCE	I/O system failure occurred during Ethernet controller chip initialization.

B.2.18 ni/crc - Cyclic Redundancy Code Test

The cyclic redundancy code (CRC) test checks the Ethernet CRC verification and bad CRC detection abilities. To run the CRC test, enter

t 3/ni/crc

and press Return.

B.2.18.1 CRC test error messages

CRC test error messages have the form

```
?TFL: 3/ni/crc (code:description)
```

- `?TFL: 3/ni/crc` indicates that the CRC test reported a problem.
- *code* represents a number that identifies the portion of the test that failed.
- *description* represents additional information that describes the failure.

Table B–12 lists the codes and descriptions used in CRC test error messages.

Table B–12: CRC Test Error Codes and Descriptions

Error Code and Description	Meaning
2: LANCE-init [x]	System could not initialize the Ethernet controller chip. The *x* represents a pass or fail code returned by one of the utilities that the test uses.

Table B–12 (Cont.): CRC Test Error Codes and Descriptions

Error Code and Description	Meaning
3: xmt [*x*]	Error occurred during packet transmission. The *x* represents a pass or fail code returned by one of the utilities that the test uses.
5: fls CRC err	Ethernet chip incorrectly flagged a good CRC as bad.
6: rcv [*x*]	Error occurred receiving a packet. The *x* represents a pass or fail code returned by one of the utilities that the test uses.
7: LANCE-init [*x*]	Error occurred when the test attempted to initialize the Ethernet controller chip. The *x* represents a pass or fail code returned by one of the utilities that the test uses.
8: xmt [*x*]	Error occurred transmitting a data packet. The *x* represents a pass or fail code returned by one of the utilities that the test uses.
10: bad CRC not dtctd	Ethernet chip did not detect a bad CRC in an incoming packet.
11: rcv [*x*]	Error occurred in packet receive operation. The *x* represents a pass or fail code returned by one of the utilities that the test uses.

B.2.19 ni/ctrs - Display Maintenance Operation Protocol (MOP) Counters Utility

The display maintenance operation protocol (MOP) counters utility displays the current MOP counters for the base system Ethernet controller. To run the MOP counters utility, enter

t 3/ni/ctrs

and press Return.

The display MOP counters utility produces no error messages.

B.2.20 ni/dma1 - Ethernet-Direct Memory Access (DMA) Registers Test

The Ethernet-direct memory access (DMA) registers test checks the Ethernet-DMA control and error registers. The test then checks the ability of the system to detect a DMA error. To run the Ethernet-DMA registers test, enter

t 3/ni/dma1

and press Return.

B.2.20.1 Ethernet-DMA registers test error messages

Ethernet-DMA registers test error messages have the form

```
?TFL: 3/ni/dmal (code: description)
```

- `?TFL: 3/ni/dmal` indicates that the Ethernet-DMA registers test reported a problem.
- *code* represents a number that identifies the portion of the test that failed.
- *description* represents additional information that describes the failure.

Table B–13 lists the codes and descriptions used in Ethernet-DMA registers test error messages.

Table B–13: Ethernet-DMA Registers Test Error Codes and Descriptions

Error Code and Description	Meaning
1: LDP wrt/rd [w=*xxxxxxxx* r=*yyyyyyyy*]	LDP register values matched when they should not. The w parameter is the value that was written to the LDP register. The r parameter is the value that was read from the LDP register.
2: LIOS wrt/rd [w=*xxxxxxxx* r=*yyyyyyyy*]	LANCE I/O slot register values matched when they should not. The w parameter is the value that was written to the LANCE I/O slot register. The r parameter is the value that was read from the LANCE I/O slot register.
3: LANCE select	LANCE I/O slot register failed to select the LANCE.
4: LANCE deselect	LANCE I/O slot register failed to deselect the LANCE.
5: err initing LANCE	Ethernet controller chip initialization failed.
6: LANCE-init timeout	Timeout occurred waiting for the LANCE initialization to finish.
7: MER	Page boundary error was not recorded in the MER register.
8: SIR	LANCE memory error bit was not set in the SIR register.
9:LANCE-start timeout	Timeout occurred waiting for LANCE to start.

B.2.21 ni/dma2 - Ethernet-Direct Memory Access (DMA) Transfer Test

The Ethernet-direct memory access (DMA) transfer test checks Ethernet DMA operation. To run the Ethernet-DMA transfer test, enter

t 3/ni/dma2

and press Return.

B.2.21.1 Ethernet-DMA transfer test error messages

Ethernet-DMA transfer test error messages have the form

```
?TFL: 3/ni/dma2 (code: description)
```

- `?TFL: 3/ni/dma2` indicates that the Ethernet-DMA transfer test reported a problem.
- *`code`* represents a number that identifies the portion of the test that failed.
- *`description`* represents additional information that describes the failure.

Table B–14 lists the codes and descriptions used in Ethernet-DMA transfer test error messages.

Table B–14: Ethernet-DMA Registers Test Error Codes and Descriptions

Error Code and Description	Meaning
2: LANCE-init [*xxxxxxxx*]	Ethernet controller chip initialization failed. *xxxxxxxx* represents a code that describes the LANCE failure.
3: xmt [*xxxxxxxx*] sz=*yyyy* ptrn=*AA*	Ethernet controller chip transmission failed. *xxxxxxxx* represents a code that describes the transmission failure. The sz parameter is the packet size. The ptrn parameter is the pattern the test tried to transmit.
4: rcv [*xxxxxxxx*] sz=*yyyy* ptrn=*AA*	Ethernet controller chip receive operation failed. *xxxxxxxx* represents a code that describes the receive failure. The sz parameter is the packet size. The ptrn parameter is the pattern the test tried to receive.
8: LANCE-DMA	DMA error occurred after a packet was received.
9: LANCE-DMA	DMA error occurred when the test began.
10: LANCE-DMA	DMA error occurred after a packet was transmitted.

B.2.22 ni/esar - Ethernet Station Address ROM (ESAR) Test

The Ethernet station address ROM (ESAR) test checks the ESAR on the Ethernet controller. To run the ESAR test, enter

t 3/ni/esar

and press Return.

B.2.22.1 ESAR test error messages

ESAR test error messages have the form

```
?TFL: 3/ni/esar (code: description)
```

- `?TFL: 3/ni/esar` indicates that the ESAR test reported a problem.
- *code* represents a number that identifies the portion of the test that failed.
- *description* represents additional information that describes the failure.

Table B–15 lists the codes and descriptions used in ESAR test error messages.

Table B–15: ESAR Test Error Codes and Descriptions

Error Code and Description	Meaning
2: ESAR[0-5] = 0's	First 6 bytes of the ESAR were 000000.
3: ESAR[0] = brdcst-sdrs	ESAR contained broadcast address.
5: checksum	ESAR checksum verificationfailed.
7: rvrs cpy !=	Reverse copy mismatch occurred.
8: frwrd cpy !=	Forward copy mismatch occurred.
9: ESAR[24-28] !=FF	Test pattern FF mismatch occurred.
10: ESAR[25-29] !=00	Test pattern 00 mismatch occurred.
11: ESAR[26-30] =55	Test pattern 55 mismatch occurred.
12: ESAR[27-31] !=AA	Test pattern AA mismatch occurred.

B.2.23 ni/ext-lb - Ethernet External Loopback Test

The external loopback test checks the Ethernet controller and its connection to the network.

Before you run the external loopback test on the base system Ethernet controller, first install a ThickWire loopback connector on the Ethernet controller. To run the external loopback test, enter

t 3/ni/ext-lb

and press Return.

B.2.23.1 External loopback test error messages

External loopback test error messages have the form

```
?TFL: 3/ni/ext-lb (code: description)
```

- `?TFL: 3/ni/ext-lb` indicates that the external loopback test reported a problem.
- *`code`* represents a number that identifies the portion of the test that failed.
- *`description`* represents additional information that describes the failure.

Table B–16 lists the codes and descriptions used in external loopback test error messages.

Table B–16: External Loopback Test Error Codes and Descriptions

Error Code and Description	Meaning
1: (LANCE-init |*xxxxxxxx*|)	LANCE initialization failed. *xxxxxxxx* represents a code that describes the LANCE failure.
3: (xmit |*xxxxxxxx*, *yyyyyyyy*| *zzzzz*)	LANCE initialization failed. *xxxxxxxx*, *yyyyyyyy* represents a code that describes the LANCE failure. *zzzzz* represents a code that describes the likely cause of the failure.
4: rcv |*xxxxxxxx*, *yyyyyyyy*|	System did not receive packet. *xxxxxxxx*, *yyyyyyyy* represents a code that describes the receive failure.
6: pkt-data !=	Transmitted packet was not received.
7	Fatal error occurred.

B.2.24 ni/int - Ethernet Interrupt Request (IRQ) Test

The interrupt request (IRQ) test checks whether the Ethernet controller can generate an interrupt to the R3000A chip. To run the IRQ test, enter

t 3/ni/int

and press Return.

B.2.24.1 IRQ test error messages

IRQ test error messages have the form

`?TFL: 3/ni/int (`*`code: description`*`)`

- `?TFL: 3/ni/int` indicates that the IRQ test reported a problem.
- *`code`* represents a number that identifies the type of failure that occurred.
- *`description`* represents additional information that describes the failure.

Table B–17 lists the codes and descriptions used in IRQ test error messages.

Table B–17: IRQ Test Error Codes and Descriptions

Error Code and Description	Meaning
1:int pndng	Pending interrupt was invalid.
2: init LANCE err = *x*	Error occurred when the system tried to initialize the Ethernet controller chip.
4: intr err xmt-stat=*x*	System generated no interrupt on packet transmission.

B.2.25 ni/int-lb - Ethernet Internal Loopback Test

The internal loopback test sends and receives data packets to and from Ethernet in internal loopback mode. To run the internal loopback test, enter

t 3/ni/int-lb

and press Return.

B.2.25.1 Internal loopback test error messages

Internal loopback test error messages have the form

```
?TFL: 3/ni/int-lb (code:description)
```

- `?TFL: 3/ni/int-lb` indicates that the internal loopback test reported a problem.
- *code* represents a number that identifies the portion of the test that failed.
- *description* represents additional information that describes the failure.

Table B–18 lists the codes and descriptions used in internal loopback test error messages.

Table B–18: Internal Loopback Test Error Codes and Descriptions

Error Code and Description	Meaning
1: rd ESAR err	The system could not access the Ethernet station address ROM.
2: LANCE-init [*xxxxxxxx*]	Error occurred initializing the Ethernet controller chip. xxxxxxxx represents a code that describes the transmission failure.
3: xmt [*xxxxxxxx*] sz=*yyyy* ptrn=*zz*	System did not transmit packet. *xxxxxxxx* represents a code that describes the transmission failure. The sz parameter is the size of the packet. The ptrn parameter is the pattern that was in the packet.
4: rcv [*xxxxxxxx*] sz=*yyyy* ptrn=*zz*	System did not receive packet. *xxxxxxxx* represents a code that describes the failure. The sz parameter is the size of the packet. The ptrn parameter is the pattern that was in the packet.
5: rcvd size=*x*, xptd=*y*	Packets received and packets sent had different sizes.
6: pkt-data !=	Data received and data sent did not match.
7	Received CRC was incorrect.

B.2.26 ni/m-cst - Ethernet Multicast Test

The multicast test checks the Ethernet ability to filter multicast packets. To run the multicast test, enter

t 3/ni/m-cst

and press Return.

B.2.26.1 Multicast test error messages

Multicast test error messages have the form

```
?TFL: 3/ni/m-cst (code: description)
```

- `?TFL: 3/ni/m-cst` indicates that the multicast test reported a problem.
- *`code`* represents a number that identifies the portion of the test that failed.
- *`description`* represents additional information that describes the failure.

Table B–19 lists the codes and descriptions used in multicast test error messages.

Table B–19: Multicast Test Error Codes and Descriptions

Error Code and Description	Meaning
1: rd ESAR err	Error occurred reading Ethernet station address ROM.
2: LANCE-init [*xxxxxxxx*]	System failed to initialize the Ethernet controller chip. *xxxxxxxx* represents a code that describes the initialization failure.
3: xmt [*xxxxxxxx*]	Ethernet controller failed to send packet. *xxxxxxxx* represents a code that describes the transmission failure.
5: rcvd invld m-cst	Ethernet controller received a multicast packet when multicast function was disabled.
6: rcv [*xxxxxxxx*]	Packet receive routine reported an error. *xxxxxxxx* represents a code that describes the receive error.
7: LANCE-init [*xxxxxxxx*]	Error occurred when the system tried to initialize the Ethernet chip. *xxxxxxxx* represents a code that describes the initialization error.
8: xmt [*xxxxxxxx*]	Ethernet controller failed to transmit packet. *xxxxxxxx* represents a code that describes the transmission error.

Table B–19 (Cont.): Multicast Test Error Codes and Descriptions

Error Code and Description	Meaning
9: rcv [*xxxxxxxx*]	Ethernet did not receive expected packet. *xxxxxxxx* represents a code that describes the receive error.

B.2.27 ni/promisc - Ethernet Promiscuous Mode Test

The promiscuous mode test checks that the Ethernet controller can receive packets in promiscuous mode. To run the promiscuous mode test, enter

t 3/ni/promisc

and press Return.

B.2.27.1 Promiscuous mode test error messages

Promiscuous mode test error messages have the form

```
?TFL: 3/ni/promisc (code: description)
```

- `?TFL: 3/ni/promisc` indicates that the promiscuous mode test reported a problem.
- *`code`* represents a number that identifies the type of failure that occurred.
- *`description`* represents additional information that describes the failure.

Table B–20 lists the codes and descriptions used in promiscuous mode test error messages.

Table B–20: Promiscuous Mode Test Error Codes and Descriptions

Error Code and Description	Meaning
2: LANCE-init [*xxxxxxxx*]	Ethernet controller initialization failed. *xxxxxxxx* represents a code that describes the initialization failure.
3: xmt [*xxxxxxxx*]	Packet transmission failed. *xxxxxxxx* represents a code that describes the transmission failure.

Table B–20 (Cont.): Promiscuous Mode Test Error Codes and Descriptions

Error Code and Description	Meaning
5: rcvd invld adrs	An inappropriate packet was received in nonpromiscuous mode.
6: rcv [*xxxxxxxx*]	Packet receive routine failed.
7: LANCE-init [*xxxxxxxx*]	System failed to initialize Ethernet controller in promiscuous mode. *xxxxxxxx* represents a code that describes the initialization failure.
8: xmt [*xxxxxxxx*]	Packet transmission failed. *xxxxxxxx* represents a code that describes the transmission failure.
9: rcv [*xxxxxxxx*]	Ethernet did not receive the expected packet while in promiscuous mode. *xxxxxxxx* represents a code that describes the receive failure.

B.2.28 ni/regs - Ethernet Registers Test

The registers test performs a read and write operation on the Ethernet registers. To run the registers test, enter

t 3/ni/regs

and press Return.

B.2.28.1 Registers test error messages

Registers test error messages have the form

```
?TFL: 3/ni/regs (code: description)
```

- `?TFL: 3/ni/regs` indicates that the registers test reported a problem.
- *`code`* represents some number that identifies the portion of the test that failed.
- *`description`* represents additional information that describes the failure.
  - The `CSR[n]` parameter, where *n* represents the number of a specific CSR register, indicates the actual value in the CSR register.
  - The `xpctd` parameter indicates the expected value for the same CSR register.

Table B-21 lists the codes and descriptions used in registers test error messages.

Table B-21: Registers Test Error Codes and Descriptions

Error Code and Description	Meaning
1: CSR[*n*]=x - xpctd 0	Write and read operation to Ethernet CSR[*n*] failed. The *n* represents the number of the CSR involved in the failure.
3: CSR[1]=*xxxx* - xpctd 0xFFFE	Writing and reading 0xFFFE failed on CSR 1. *xxxx* represents the actual value in CSR 1.
4: CSR[2]=*xxxx* - xpctd 0x00FF	Writing and reading 0x00FF failed on CSR 2. *xxxx* represents the actual value in CSR 2.
5: CSR[1]=*xxxx* - bit lk frm CSR[2]	Bit leak from CSR2 to CSR1 occurred. *xxxx* represents the actual value in CSR[1].
6: CSR[2]= *xxxx* - bit lk frm CSR[1]	Bit leak from CSR1 to CSR2 occurred. *xxxx* represents the actual value in CSR[2].
7: immediate write/read flr	Immediate write and read failure occurred.

B.2.29 prcache - Prcache Quick Test

The prcache quick test of NVRAM on power-up tests the scratch area of the optional NVRAM module. The diagnostic status bit in the diagnostic register on the NVRAM module is set on failure. The optional NVRAM module can be installed in memory slot 14, and is different from the system module NVR that is tested by the rtc/nvr test. To run the prcache quick test, enter

t 3/prcache

and press Return.

For a thorough test, first zero the NVRAM memory by entering

t 3/prcache/clear

and press Return.

B.2.29.1 Prcache quick test error messages

Prcache quick test error messages have the form

```
?TFL: 3/prcache (code: description)
```

- `?TFL: 3/prcache` indicates that the prcache quick test reported a problem.

- `code` represents a number that identifies the portion of the test that failed.
- `description` represents additional information that describes the failure.

Table B–22 lists the codes and descriptions used in registers test error messages.

Table B–22: Prcache Quick Test Error Codes and Descriptions

Error Code and Description	Meaning
1: board 14: MBE = *X* SBE = *Y*	*X* multiple bit errors and *Y* single bit errors occurred on the NVRAM board in slot 14.
2: board 14, too many SBEs: *X*	Too many single bit errors. *X* single bit errors occurred on the NVRAM board in slot 14.

B.2.30 prcache/arm - Disconnect Battery Command

The prcache/arm command turns off the battery on the NVRAM module. To run the prcache/arm command, enter

t 3/prcache/arm [*board*]

and press Return.

When you enter the prcache/arm command, replace the optional *board* parameter with the slot number of the NVRAM module. On the DECstation 5000 Model 240, the NVRAM module must always be installed in slot 14. The default value is 14.

B.2.30.1 Prcache/arm command error message

If the prcache/arm command does not complete successfully, it returns the following error message:

```
?TFL: 3/prcache/arm (1:(tst nocomp))
```

B.2.31 prcache/clear - Zero NVRAM Memory Command

The prcache/clear command quickly writes zeros to all NVRAM memory addresses. To run the prcache/clear command, enter

t 3/prcache/clear [*board*]

and press Return.

When you enter the prcache/clear command, replace the optional *board* parameter with the slot number of the NVRAM module. On the DECstation 5000 Model 240, the NVRAM module must always be installed in slot 14. The default value is 14.

If the prcache contains valid data, the system responds with the following prompt:

```
prcache valid data - wrt ? (1/0)
```

enter **1** to clear the cache. Enter **0** to cancel the prcache/clear command.

B.2.31.1 Prcache clear error message

If the prcache/clear command does not complete successfully, it returns the following error message:

```
?TFL: 3/prcache/clear (1:(tst nocomp))
```

B.2.32 prcache/unarm - Connect Battery Command

The prcache/unarm command turns on the battery on the NVRAM module. To run the prcache/unarm command, enter

t 3/prcache/unarm [*board*]

and press Return.

When you enter the prcache/unarm command, replace the optional *board* parameter with the slot number of the NVRAM module. On the DECstation 5000 Model 240, the NVRAM module must always be installed in slot 14. The default value is 14.

The prcache/unarm command returns no error message.

B.2.33 rtc/nvr - Nonvolatile RAM Test

The nonvolatile RAM (NVR) test checks the system module nonvolatile RAM. The system module NVR is different from the optional NVRAM cache module that can be installed in memory slot 14 and that is tested by the prcache test. To run the NVR test, enter

t 3/rtc/nvr [*pattern*]

and press Return.

When you enter the NVR test command, replace the optional *pattern* parameter with a specific pattern that you want to use in the test. The default pattern is 55.

B.2.33.1 NVR test error messages

NVR test error messages have the form

```
?TFL: 3/rtc/nvr (code:  address=actual,
sb expected)
```

- `?TFL 3/rtc/nvr` indicates that the NVR test read an incorrect pattern from the NVR.
- *`code`* represents a number that identifies the portion of the test that failed.
- *`address`* represents the address at which the error occurred.
- *`actual`* represents the value at that address.
- *`expected`* represents the expected value at that address.

B.2.34 rtc/period - Real-Time Clock Period Test

The real-time clock (RTC) period test checks the RTC periodic interrupt operation. To run the RTC period test, enter

t 3/rtc/period

and press Return.

B.2.34.1 RTC period test error messages

RTC period test error messages have the form

```
?TFL: 3/rtc/period/(code)
```

- `?TFL 3/rtc/period` indicates that the RTC period test reported an error.
- *`code`* represents a number that identifies the portion of the test that failed.

Table B–23 lists the codes used in RTC period test error messages.

Table B–23: RTC Period Test Error Codes

Error Code	Meaning
1	Update-in-progress (UIP) bit remained set past the allotted time.
2	Real-time clock interrupt was pending when it should not have been.
3	Allowed time ran out while waiting for interrupt.

B.2.35 rtc/regs - Real-Time Clock Registers Test

The real-time clock registers test checks the real-time clock (RTC) registers. To run the real-time clock registers test, enter

t 3/rtc/regs

and press Return.

B.2.35.1 Real-time clock registers test error messages

Real-time clock register test error messages have the form

`?TFL: 3/rtc/regs (`*`code: description`*`)`

- `?TFL 3/rtc/regs` indicates that the real-time clock register test reported an error.
- *`code`* represents a number that identifies the portion of the test that failed.
- *`description`* represents additional information that describes the failure.

Table B–24 lists the codes and descriptions used in real-time clock register test error messages.

Table B–24: Real-Time Clock Register Test Error Codes and Desscriptions

Error Code and Description	Meaning
1	UIP bit remained set past allotted time.
2: register=*actual*, sb *expected*	The test failed to write pattern correctly. The register value is the actual value in the named register, followed by the expected value.

B.2.36 rtc/time - Real-Time Test

The real-time test checks times generated by the real-time clock against hard-coded time values. To run the real-time test, enter

t 3/rtc/time

and press Return.

B.2.36.1 Real-time test error messages

Real-time test error messages have the form

```
?TFL: 3/rtc/time (code)
```

- `?TFL 3/rtc/time` indicates that the real time test reported an error.
- *code* represents a number that identifies the portion of the test that failed.

Table B–25 lists the codes used in real-time test error messages.

Table B–25: Real-Time Test Error Codes

Error Code	Meaning
1	UIP bit remained set past allotted time.
2	Real-time clock interrupt was pending when it should not have been.
3	Allowed time ran out while waiting for interrupt.
4	Allowed time ran out while waiting for second interrupt.
5	UIP bit remained set past second allotted time.
6	Real-time clock seconds were not set to 0 on wraparound.
7	Real-time clock minutes were not set to 0.
8	Real-time clock hours were not set to 0.
9	Real-time clock day-of-the-week was not set to 1.
10	Real-time clock date was not set to 1.
12	Real-time clock month was not set to 1.
13	Real-time clock year was not set to 0.

B.2.37 scc/access - Serial Communication Chip (SCC) Access Test

The serial communication chip (SCC) access test checks whether the system can perform a read and write operation on the SCC. To run the SCC access test, enter

t 3/scc/access

and press Return.

B.2.37.1 SCC access test error messages

The only SCC access test error message is

```
?TFL: 3/scc/access (1:LnM reg-N:
actual=0xXX xpctd=0xYY
```

- `?TFL: 3/scc/access` indicates that the read and write operation on the SCC failed.
- *M* represents the number of the serial line where the error occurred.
- *N* represents the number of the register that failed the test.
- The `actual` value is the value in that register.
- The `xpctd` value is the expected value for that register.

B.2.38 scc/dma - Serial Communication Chip Direct Memory Access Test

The serial communication chip (SCC) direct memory access (DMA) test checks the ability of the serial communication and IO-ASIC chips to perform a DMA operation. To run the SCC DMA test, enter

t 3/scc/dma [*line*] [*loopback*] [*baud*]

and press Return.

- Replace the optional *line* parameter with a value that specifies the line to test.
  - Specify 2 to test serial line number 2. The default value is 2.
  - Specify 3 to test serial line number 3.
- Replace the optional *loopback* parameter with a value that specifies the type of loopback operation the test is to perform.
  - Specify intl to run an internal loopback operation. The default value is intl.
  - Specify extl to run an external loopback operation.
- Replace *baud* with the baud rate at which you want the test to run. You can specify one of the following baud rates:
  300
  4800
  1200
  9600
  2400

19200
38400

The default value is 38400.

B.2.38.1 SCC DMA test error messages

SCC DMA test error message have the form

```
?TFL: 3/scc/dma code:LnN
SIR_xptd=xxxxxxxx SIR=yyyyyyyy
SSR=zzzzzzzz
```

- `?TFL: 3/scc/dma` indicates that the SCC DMA test reported an error.
- *code* represents a number that identifies the part of the test that failed.
- *N* represents the number of the serial line that reported the error.
- The `SIR_xpted` value is the expected value for the system interrupt register.
- The `SIR` value is the actual value in the system interrupt register.
- The `SSR` value is the value in the system status register.

Table B–26 lists the codes used in SCC DMA test error messages.

Table B–26: SCC DMA Test Error Codes

Error Code	Meaning
1	SIR values are invalid.
2	Miscompare occurred during DMA read and write operation.
3	Overrun occurred in the receive buffer.
4	Interrupt signal was not sent to the system.

B.2.39 scc/int - Serial Communication Chip Interrupts Test

The serial communication chip (SCC) interrupts test checks the ability of the SCC to perform internal, external, and countdown interrupts. To run the SCC interrupts test, enter

t 3/scc/int [*line*]

and press Return. Replace the optional *line* parameter with a value that specifies which line to test.

B.2.39.1 SCC interrupts test error messages

SCC interrupts test error messages have the form

```
?TFL: 3/scc/int (code: lnN
RR0=xx RR3=yy
SIR=zzzzzzzz)
```

- `?TFL: 3/scc/int` indicates that the SCC interrupts test reported an error.
- *code* represents a number that indicates which portion of the test reported the error.
  - If the number is odd, the bits to set the interrupt to on were invalid.
  - If the number is even, the bits to set the interrupt to off were invalid.
- *N* represents the number of the serial line where the error occurred.
- The `RR0` value is the contents of the SCC read register 0.
- The `RR3` value is the contents of the SCC read register 3.
- The `SIR` value is the contents of the system interrupt register.

B.2.40 scc/io - Serial Communication Chip Input/Output (I/O) Test

The serial communication chip (SCC) input/output (I/O) test checks the ability of the SCC to perform an I/O operation on a serial line. To run the SCC I/O test, enter

t 3/scc/io [*line*] [*loopback*]

and press Return.

When you enter the SCC I/O test command,

- Replace the optional *line* parameter with a value that specifies the line to test.
  - Specify 0 to test serial line 0. The default value is 0.
  - Specify 1 to test serial line 1.
  - Specify 2 to test serial line 2.
  - Specify 3 to test serial line 3.

- Replace the optional *loopback* parameter with a value that specifies the type of loopback operation the test performs.
  - Specify intl to run an internal loopback operation. The default value is intl.
  - Specify extl to run an external loopback operation.

B.2.40.1 SCC I/O test error messages

SCC I/O test error messages have the form

```
?TFL: 3/scc/io (code: LnNdescription)
```

- `?TFL: 3/scc/io` indicates that the SCC I/O test reported an error.
- *`code`* represents a number that identifies the portion of the test that failed.
- *`N`* represents the number of the line in which the error occurred.
- *`description`* represents additional information that describes the error.

Table B–27 lists the codes and descriptions used in SCC I/O test error messages.

Table B–27: SCC I/O Test Error Codes and Descriptions

Error Code and Description	Meaning
1: Ln*N* tx bfr not empty. status=*xx*	System could not write a single character because the transmit buffer was not empty. *N* represents the line in which the error occurred. The status value is the contents of read register 0.
2: Ln*N* char not rcvd. status=*x*	CHAR AVAIL signal not received when the system was expecting a character. *N* represents the line in which the error occurred. The status value is the contents of read register 0.
3: Ln*N* expctd=*xx*, rcvd=*yy*, status=*zz*	The character that was received was different than the transmitted character. *N* represents the line in which the error occurred. *xx* represents the transmitted value. *yy* represents the received value. The status value is the contents of read register 0.

B.2.41 scc/pins - Serial Communication Chip Pins Test

The serial communication chip (SCC) pins test checks the control pins on the communications connectors. To run the SCC pins test, enter

t 3/scc/pins [*line*] [*loopback*]

and press Return.

- Replace the optional *line* parameter with a value that specifies the communications connector that you want to test.
  - Specify 2 to test the communications connector on the right as you face the back of the system unit.
  - Specify 3 to test the communications connector on the left as you face the back of the system unit.
- Replace the optional *loopback* parameter with a value that specifies the loopback hardware that you attach to the communications connector being tested.
  - Specify 29-24795 if you attach a 29-24795 loopback connector. The default value is 29-24795.
  - Specify H8571 if you attach an H8571 loopback connector.
  - Specify hm if you attach an hm loopback connector.
  - Specify H3200 if you attach an H3200 loopback connector.

Table B–28 lists the specific pin pairs that each loopback connector tests.

Table B–28: Pin Pairs Tested by Individual Loopback Connectors

Loopback connector	Pin pairs tested	Meaning
29-24795	4-5	RTS[1] to CTS[2]
	23-6-8	SS[3] to DSR[4] and CD[5] 6-23 failure implies 6 broken. 8-23 failure implies 8 broken. 6-23 8-23 failure implies 23 broken.

[1]Request to send
[2]Clear to send
[3]Secondary request to send
[4]Data set ready
[5]Carrier detector

Table B–28 (Cont.): Pin Pairs Tested by Individual Loopback Connectors

Loopback connector	Pin pairs tested	Meaning
H3200	4-5	RTS to CTS
	6-20	DSR to DTR[6]
	12-23	SI[7] to SS
H8571-A	4-5	RTS to CTS
	20-6-8	DTR to DSR and CD 6-20 failure implies 6 broken. 8-20 failure implies 8 broken. 6-20 8-20 failure implies 20 broken.
hm	4-5	RTS to CTS

[6]Data terminal ready

[7]Secondary signal

B.2.41.1 SCC pins test error messages

SCC pins test error message have the form

```
?TFL: 3/scc/pins (code: LnN:  description)
```

- `?TFL: 3/scc/pins` indicates that the SCC pins test reported an error.
- *N* represents the number of the serial line in which the error occurred.
- *description* represents additional information that describes the error.

Table B–29 lists the codes and descriptions used in SCC pins test error messages.

Table B–29: SCC Pins Test Error Codes and Descriptions

Error Code and Description	Meaning
1:Ln*N* Invld param [*xxxxx*]	The number used in the test command to specify the loopback was invalid. *N* represents the number of the serial line in which the error occurred. *xxxxx* represents the first two characters of the invalid value that was specified.
2:Ln*N* Strtup R-*xx* xptd=*yy* actl=*zz* | pins |	Test failed to generate the expected SCC status bits. *N* represents the number of the serial line in which the error occurred. The Strtup R value is the number of SCC register that contains the status bits. The xptd value is the expected status bits. The actl value is the actual status bits. *pins* represents the pin pairs for which the test was set up.
3: Ln*N* *xxxxx*	Pins failed to respond properly. *xxxxx* represents the numbers of one or more pin pairs that failed the test.

B.2.42 scc/tx-rx - Serial Communication Chip Transmit and Receive Test

The serial communication chip (SCC) transmit and receive test checks the ability of the SCC to transmit and receive information. To run the SCC transmit and receive test, enter

t 3/scc/tx-rx [*line*] [*loopback*] [*baud*]) [*parity*]) [*bits*]

and press Return.

- Replace the optional *line* parameter with a value that specifies the serial line to test.
  - — Specify 0 to test serial line 0. The default value is 0.
  - — Specify 1 to test serial line 1.
  - — Specify 2 to test serial line 2.
  - — Specify 3 to test serial line 3.
- Replace the optional *loopback* parameter with a value that specifies the type of loopback operation the test performs.
  - — Specify intl to run an internal loopback operation. The default value is intl.
  - — Specify extl to run an external loopback operation.

- Replace the optional *baud* parameter with a value that specifies the baud rate at which the test runs. You can specify one of the following baud rates:
  300
  1200
  2400
  3600
  4800
  9600
  19200

  The default baud rate is 9600.
- Replace the optional *parity* parameter with a value that specifies the type of parity that the test uses.
  - Specify none to use no parity. The default value is none.
  - Specify odd to use odd parity.
  - Specify even to use even parity.
- Replace the optional *bits* parameter with a value that specifies the number of bits per characters that the test uses.
  - Specify 8 to use 8 bits per character. The default value is 8.
  - Specify 7 to use 7 bits per character.
  - Specify 6 to use 6 bits per character.

B.2.42.1 SCC transmit and receive test error messages

SCC transmit and receive test error message have the form

`?TFL: 3/scc/tx-rx (code:LnN description)`

- `?TFL: 3/scc/tx-rx` indicates that the SCC transmit and receive test failed.
- `code` represents a number that indicates the portion of the test that failed.
- *`description`* represents additional information that describes the error.

Table B–30 lists the codes and descriptions used in SCC transmit and receive error messages.

Table B–30: SCC Transmit and Receive Test Error Codes and Descriptions

Error Code and Description	Meaning
1: Ln*N* tx bfr not empty. status=*xx*	System could not write a single character because the transmit buffer was not empty. *N* represents the line in which the error occurred. The status value is the contents of SCC read register 0.
2: Ln*N* char not rcvd. status=*xx*	CHAR AVAIL signal not received when the system was expecting a character. *N* represents the line in which the error occurred. The status value is the contents of SCC read register 0.
3: Ln*N* expctd=*xx*, rcvd=*yy*, status=*zz*	The character that was received was different than the transmitted character. *N* represents the line in which the error occurred. The *xx* represents the transmitted value. The *yy* represents the received value. The status value is the contents of SCC read register 0.
4: Ln*N* Rx err. errs=*xx*	Receiving character in FIFO reported an error. *N* represents the line in which the error occurred. The errs value equals the number of associated input character FIFO error bits.

B.2.43 scsi/cntl - SCSI Controller Test

The SCSI controller test checks SCSI controller operation. To run the SCSI controller test, enter

t *slot_number*/**scsi/cntl**

and press Return. Replace *slot_number* with the slot number of the SCSI controller to be tested.

B.2.43.1 SCSI controller test error messages

SCSI controller test error messages have the form

```
? TFL: 3/scsi/cntl (code: description)
```

- `?TFL 3/scsi/cntl` indicates that the SCSI controller test failed on the base system module SCSI controller.
- `code` represents a number that indicates the portion of the test that failed.
- `description` represents additional information that describes the failure.

Table B–31 lists the error descriptions used in SCSI controller test error messages.

Table B–31: SCSI Controller Error Codes and Descriptions

Error Code and Description	Meaning
1: rd cnfg	Values written to and read from configuration register did not match.
2: fifo flg	First in, first out (FIFO) load and FIFO flags did not match.
3: cnt xfr	Write and read operation for TCL register reported a mismatch.
4: illg cmd	Command was illegal and did not generate an interrupt.
5: int reg	Controller cannot clear internal interrupt register.
6: rd cnfg	Mismatch occurred when reading the write/read configuration register.

B.2.44 scsi/sdiag - SCSI Send Diagnostics Test

The SCSI send diagnostics test runs the self-test for an individual SCSI device. You can specify whether the test alters drive parameters and includes a write operation. To run the SCSI send diagnostics test, enter

t *slot_number*/**scsi/sdiag [d] [u] [s]**

and press Return.

- Replace *slot_number* with the slot number of the module to be tested.
- Replace *scsi_id* with the SCSI ID of the device you want to test. The default value is 0.
- Include the optional d and u parameters to specify the conditions that those parameters set for the specific drive you are testing. Note that the result of including d and u depends on the specific drive. To determine the effect of including d and u, refer to the service guide for the drive that you want to test.
- Include the optional s parameter to suppress the error message display.

B.2.44.1 SCSI send diagnostics test error messages

SCSI send diagnostic test error messages have the form

```
?TFL: 3/sdiag code: description
```

- `?TFL 3/scsi/sdiag` indicates that the SCSI send diagnostics test reported an error on the base system module SCSI controller.

- `code` represents a number that indicates the portion of the test that failed.
- `description` represents additional information that describes the failure.

Table B–32 lists the codes and descriptions used in SCSI send diagnostics test error messages.

Table B–32: SCSI Send Diagnostics Test Error Descriptions

Error Description	Meaning
1: dev ol	Test could not bring the unit on line.
2: dev ol	Test could not bring the unit on line.
3: sdiag	Device failed the send diagnostics test.

B.2.45 scsi/target - SCSI Target Test

The SCSI target test performs a read test on a specific SCSI device. If you include the optional write parameter, the test also performs a write test. To run the SCSI target test, enter

t 3/scsi/target *scsi_id* [**w**] [**l***loops*]

and press Return.

- Replace the *scsi_id* parameter with the SCSI ID of the device you want to test.
- Specify the optional w parameter to include a write operation in the SCSI target test.
- Specify the optional l parameter to have the test repeat up to 9 times. If you include the l parameter, replace *loops* with the number of times you want the test to repeat.

CAUTION: *This test can destroy existing data if it is run with the w option. The test writes over existing data at random.*

B.2.45.1 SCSI target test error messages

SCSI target test error messages have the form

```
?TFL: 3/scsi/target (code: description)
```

- `?TFL 3/scsi/target` indicates that the SCSI target test reported an error.

- `code` represents a number that indicates the portion of the test that failed.
- `description` represents additional information that describes the failure.

CAUTION: *Always follow antistatic procedures when handling electronic components.*

Table B–33 lists the codes and descriptions used in SCSI target test error messages.

Table B–33: SCSI Target Test Error Codes and Descriptions

Error Code and Description	Meaning
1: (dev ol) *N*	Test could not bring the device on line. *N* represents the SCSI ID of the device that could not be brought on line.
2: (tst nocomp) *N*	Command entered from the keyboard aborted the test. *N* represents the SCSI ID of the device being tested.
3: (ro dev) *N*	Test cannot perform write operation. Device is a read-only device. *N* represents the SCSI ID of the device specified in the test.
4: (dev type) *N*	Test does not test the specified device. *N* represents the SCSI ID of the device specified in the test.
6: (rdCap*N*	Read capacity command failed. *N* represents the SCSI ID of the device that could not be brought on line.
7: (rzWr) *N*	Write operation failed. *N* represents the SCSI ID of the device that failed.
8: (rzRd) *N*	Read operation failed. *N* represents the SCSI ID of the device that failed.
9: (cmp) *N*	Write and read values did not match. *N* represents the SCSI ID of the device involved in the miscompare.
10: (wrFlMrk) *N*	Write file mark failed. *N* represents the SCSI ID of the device being tested.
11: (tzWr) *N*	Write operation failed. *N* represents the SCSI ID of the device that failed.
12: (wrFlMrk) *N*	Write file mark failed. *N* represents the SCSI ID of the device being tested.
13: (spc) *N*	Space (-2) operation failed. *N* represents the SCSI ID of the device involved in the failure.
14: (spc) *N*	Space (1) operation failed. *N* represents the SCSI ID of the device involved in the failure.
15: (tzRd) *N*	Read operation failed. *N* represents the SCSI ID of the device being tested.
16: (cmp) *N*	Write and read values did not match. *N* represents the SCSI ID of the device involved in the miscompare.

B.2.46 tlb/prb - Translation Lookaside Buffer Probe Test

The translation lookaside buffer (TLB) probe test checks whether all TLB registers respond to an address match operation. To run theTLB probe test, enter

t 3/tlb/prb

and press Return.

B.2.46.1 TLB probe test error messages

The only TLB probe test error message is

`?TFL: 3/tlb/prb (match(0, N)=actual, sb expected)`

- `?TFL: 3/tlb/prb (match(0,N))` indicates that the value at address 0 did not match the value at the address represented by *N*.
- *`actual`* represents the actual value found at the address represented by *N*.
- *`expected`* represents the expected value at the address represented by *N*.

B.2.47 tlb/reg - Translation Lookaside Buffer Registers Test

The translation lookaside buffer (TLB) registers test performs a read and write operation on the TLB. To run the TLB registers test, enter

t 3/tlb/reg [*pattern*]

and press Return.

Replace the optional *pattern* parameter with the pattern you want to use for the read and write operation. The default pattern is 55555555.

B.2.47.1 TLB registers test error messages

TLB registers test error messages have the form

`?TFL: 3/tlb/regs (description)`

- `?TFL 3/tlb/regs` indicates that the TLB registers test reported an error.
- *`description`* represents additional information that describes the error.

Table B–34 lists the descriptions used in TLB registers test error messages.

Table B–34: TLB Registers Test Error Descriptions

Error Description	Meaning
tlblo [*N*]= *actual*, sb *expected*	Pattern in TLB low (LO) register was not the expected pattern. *N* represents the number of the register with the incorrect value. The actual and expected values follow.
tlbhi [*N*]= *actual*, sb *expected*	Pattern in TLB high (HI) register was not the expected pattern. *N* represents the number of the register with the incorrect value. The actual and expected values in the register follow.

Appendix C
CPU and System Registers

This appendix describes the CPU and system registers. The CPU and system registers contain information that can be useful when troubleshooting.

There are two types of registers: CPU registers and system registers. The system automatically displays CPU register information on the screen when exceptions occur. Use the e command in console mode to access system registers.

C.1 CPU Registers

Table C–1 lists the CPU registers.

Table C–1: CPU Registers

Register	Description
Cause	Cause of last exception
EPC	Exception program counter
Status	Status register
BadVAddr	Bad virtual address (read only)

When an exception occurs, the system automatically displays CPU register information in one of two formats.

The first format is as follows:

```
?TFL slot_number/test_name
(CUX, cause= xxxxxxxx) [KN03-GA]

?TFL slot_number/test_name
(UEX, cause= xxxxxxxx) [KN03-GA]
```

where

slot_number represents the slot number of the module being tested.
test_name represents the name of the test being run.

xxxxxxxx represents the contents of the cause register.

The second format is as follows:

```
? PC:   0x451<vtr=nrml>
? CR:   0x810<ce=0,ip4,exc=AdEL>
? SR:   0x30080000<cu1,cu0,cm,ipl=8>
? VA:   0x0x451
? ER:   0x100003f0
? MER: 0x2000
```

where the values on each line are as follows:

PC = Address of the exception instruction
CR = Contents of the cause register
SR = Contents of the status register
VA = Virtual address of the exception
ER =Contents of the error address register
MER =Contents of the memory error register

Refer to Chapter 9 for detailed troubleshooting information.

C.1.1 Cause Register

The cause register is a 32-bit read/write register that describes the nature of the last exception. A 4-bit exception code indicates the cause of the exception, and the remaining fields contain detail information relevant to the handling of certain types of exceptions.

The branch delay (BD) bit indicates whether the exception program counter (EPC) was adjusted to point at the branch instruction that precedes the next restartable instruction. For a coprocessor unusable exception, the coprocessor error (CE) field indicates the coprocessor unit number referenced by the instructions that caused the exception.

The interrupt pending (IP) field indicates which external, internal, coprocessor, and software interrupts are pending. You can write to $IP_{1..0}$ to set or reset software interrupts. The remaining bits, $IP_{7..2}$, are read only and represent external, internal, or coprocessor interrupts.

The number and assignment of the IP bits are implementation-dependent. R3000A processors have six external interrupts. IP5 is used for the MIPS floating-point coprocessor interrupt. IP2 is normally used for system bus (I/O) interrupts. The cause register has the following format:

```
  31    30    29  28 27                                 16
+-----+-----+------+--------------------------------+
| BD  |  0  |  CE  |              0                 |
+-----+-----+------+--------------------------------+
   1     1      2               12

 15              10 9    8 7     6 5        2 1    0
+-----------------+------+------+-----------+-----+
|        IP       |  SW  |  0   |  ExcCode  |  0  |
+-----------------+------+------+-----------+-----+
          6          2      2         4        2
```

- BD indicates whether the last exception occurred during execution in a branch delay slot (0 = normal, 1 = delay slot).
- CE indicates the coprocessor unit number reference when a coprocessor unusable exception occurs.
- IP indicates whether an interrupt is pending.
- SW indicates which of two software interrupts is pending.
- ExcCode is the exception code field. Table C–2 lists the exception codes and their meanings.
- 0 is unused (ignored on write, zero when read).

Table C–2: Exception Codes

Code Number	Mnemonic	Description
0	Int	Interrupt
1	Mod	TLB modification exception
2	TLBL	TLB miss exception (load or instruction fetch)
3	TLBS	TLB miss exception (store)
4	AdEL	Address error exception (load or instruction fetch)
5	AdES	Address error exception (store)
6	IBE	Bus error exception (instruction fetch)
7	DBE	Bus error exception (data reference: load or store)
8	Sys	Syscall exception
9	Bp	Breakpoint exception
10	RI	Reserved instruction exception
11	CpU	Coprocessor unusable exception
12	OV	Arithmetic overflow exception
13-31		Reserved

C.1.2 Exception Program Counter (EPC) Register

The exception program counter (EPC) register indicates the virtual address at which the most recent exception occurred. This register is a 32-bit read-only register that contains an address at which instruction processing can resume after an exception is serviced. For synchronous exceptions, the EPC register contains the virtual address of the instruction that was the direct cause of the exception. When that instruction is in a branch delay slot, the EPC register contains the virtual address of the immediately preceding branch or jump instruction.

If the exception is caused by recoverable, temporary conditions (such as a TLB miss), the EPC register contains a virtual address at the instruction that caused the exception. Thus, after correcting the conditions, the EPC registers contains a point at which execution can be legitimately resumed. The EPC register has the following format:

```
 31                                              0
+------------------------------------------------+
|                      EPC                        |
+------------------------------------------------+
                       32
```

EPC is the exception program counter.

C.1.3 Status Register

The status register (SR) is a 32-bit read/write register that contains the kernel/user mode, interrupt enable, and diagnostic state of the processor. The SR contains a three-level stack (current, previous, and old) of the kernel /user (KU) bit and the interrupt enable (IE) bit. The stack is pushed when each exception is taken. The stack is popped by the restore from exception (RFE) instruction. These bits can also be directly read or written.

The status register has the following format:

```
 31                         28 27                     16
+----------------------------+-----------------------+
|           CU               |          DS           |
+----------------------------+-----------------------+
            18                         12

 15      8 7    6    5     4     3     2     1     0
+---- ---+-----+-----+-----+-----+-----+-----+-----+
|   IM   |  0  | KUo | IEo | KUp | IEp | KUc | IEc |
+--------+-----+-----+-----+-----+-----+-----+-----+
    8       2     1     1     1     1     1     1
```

- The coprocessor usability (CU) field is a 4-bit field that individually controls the usability of each of the four coprocessor unit numbers (1 = usable, 0 = unusable). Coprocessor zero is always usable in kernel mode, regardless of the setting of the CU0 bit.
- The diagnostic status (DS) field is an implementation-dependent 12-bit diagnostic status field that is used for self-testing and checking the cache and virtual memory system. For a detailed description of the DS field, see the "Diagnostic Register" section in this appendix
- The interrupt mask field (IM) is an 8-bit field that controls the enabling of each of eight external interrupt conditions. It controls the enabling of each of the external, internal, coprocessor, and software interrupts (0 = disable, 1 = enable). If interrupts are enabled, an external interrupt occurs when corresponding bits are set in both the interrupt mask field of the SR and the interrupt pending (IP) field of the cause register. The actual width of this register is machine-dependent. For a description of the IP field, see the "Cause Register" section in this appendix.

- KUo is the old kernel/user mode (0 = kernel, 1 = user).
- IEo is the old interrupt enable setting (0 = disable, 1 = enable).
- KUp is the previous kernel/user mode (0 = kernel, 1 = user).
- IEp is the previous interrupt enable setting (0 = disable, 1 = enable).
- KUc is the current kernel/user mode (0 = kernel, 1 = user).
- IEc is the current interrupt enable setting (0 = disable, 1 = enable).

C.1.3.1 Diagnostic status

The diagnostic facilities depend on the characteristics of the cache and virtual memory system of the implementation. Therefore, the layout of the diagnostic status field is implementation-dependent. The diagnostic status field is normally used for diagnostic code and, in certain cases, for operating system diagnostic facilities (such as reporting parity errors). On some machines it is used for relatively rare operations such as flushing caches. Normally, this field should be set to 0 by operating system code. The diagnostic status bits are BEV, TS, PE, CM PZ, SwC, and IsC. This set of bits provides a complete fault detection capability, but is not intended to provide extensive fault diagnosis.

The diagnostic status field has the following format:

```
 27          23  22    21   20   19   18   17    16
+------------+-----+----+----+----+----+-----+-----+
|     0      | BEV | TS | PE | CM | PZ | SwC | IsC |
+------------+-----+----+----+----+----+-----+-----+
      5         1     1    1    1    1    1     1
```

- BEV controls the location of UTLB miss and general exception vectors (0 = normal, 1 = bootstrap). When this bit is set, the UTLB miss exception vector is relocated to address 0xbfc00100 and the general exception vector is relocated to address 0xbfc00180 (general).
- TS indicates that TLB shut down occurred.
- PE indicates that a cache parity error occurred. This bit can be cleared by writing 1 to this bit position.
- CM indicates whether a data cache miss occurred while the system was in cache test mode (0 = hit, 1 = miss).

- PZ controls the zeroing of cache parity bits (0 = normal, 1 = parity forced to zero).
- SwC controls the switching of the data and instruction caches (0 = normal, 1 = switched).
- IsC controls isolation of the cache (0 = normal, 1 = cache isolated).
- 0 is unused (ignored on write, zero when read).

C.1.4 BadVAddr Register

The bad virtual address (BadVAddr) register is a 32-bit read-only register that contains the most recently translated virtual address for which a translation error occurred.

The bad virtual address register has the following format:

```
+---------------------------------------+
|               BadVAddr                |
+---------------------------------------+
```

BadVAddr is the bad virtual address.

C.2 System Registers

This section describes the system registers used for troubleshooting.

C.2.1 Data Buffers 3 to 0

The data buffers are general-purpose 32-bit read-write registers used by the I/O control ASIC. These registers can be read and written for test purposes. Any direct memory access (DMA) or access to a peripheral device can overwrite these registers. To ensure proper testing, disable all DMA engines.

Table C–3 lists the system registers.

Table C–3: System Registers

Register	Console Address	Description
SSR	0xBF840100	System support register
SIR	0xBF840110	System interrupt register
Mask	0xBF840120	System interrupt mask register

Table C–3 (Cont.): System Registers

Register	Console Address	Description
EAR	0xBFA40000	Error address register
ES	0xBFA80000	Memory error check/syndrome register
CS	0xBFAC0000	Memory bank size and ECC diagnostics register

Use the e command to examine the contents of a system register from the console. Enter the e command in the following format:

e [*options*] [*console_address*]

See Appendix C, "Console Commands," for information about the formats and options used with the e command and the other console commands.

C.2.2 System Support Register (SSR)

The system support register (SSR) can be both read from and written to. Bits <31:16> are used internally by the I/O control ASIC. Bits <15:0> generate signals visible outside the I/O control ASIC.

Table C–4: System Support Register 0xBF840100

Bits	Access	Description
31	R/W	Communication port 1 transmit DMA enable (1=enable, 0=disable)
30	R/W	Communication port 1 receive DMA enable (1=enable, 0=disable)
29	R/W	Communication port 2 transmit DMA enable (1=enable, 0=disable)
28	R/W	Communication port 2 receive DMA enable (1=enable, 0=disable)
27:23	R/W	Reserved.
22	R/W	Reserved
21	R/W	Reserved
20	R/W	Reserved
19	R/W	Reserved
18	R/W	SCSI DMA direction, 0 = transmit (read from memory)
17	R/W	SCSI DMA enable (1=enable, 0=disable)
16	R/W	LANCE DMA enable (1=enable, 0=disable)

Table C-4 (Cont.): System Support Register 0xBF840100

Bits	Access	Description
15	R/W	DIAGDN (diagnostic flag)
14:13	R/W	TXDIS (serial transmit disable)
12	R/W	Reserved
11	R/W	SCC reset (active low)
10	R/W	RTC reset (active low)
9	R/W	53C94 SCSI controller reset (SCSI active low)
8	R/W	LANCE reset (Ethernet active low)
7..0	R/W	LEDs

SSR<31>

When set to 1, this bit enables communication port 1 (serial line 2) to transmit DMA to SCC(0)-B. Communication port 1 is the right comm port, viewed from the back.

SSR<30>

When set to 1, this bit enables communication port 1 (serial line 2) to receive DMA from SCC(0)-B. Communication port 1 is the right comm port, viewed from the back.

SSR<29>

When set to 1, this bit enables communication port 2 (serial line 3) to transmit DMA to SCC(1)-B. Communication port 2 is the left comm port, viewed from the back.

SSR<28>

When set to 1, this bit enables communication port 2 (serial line 3) to receive DMA from SCC(1)-B. Communication port 2 is the left comm port, viewed from the back.

SSR<27:19>

These bits are reserved.

SSR<18>

This bit, set to 0 on power-up or reset, determines the direction of the SCSI DMA transfer. If the bit is 0, memory data will be supplied to the 53C94 SCSI controller upon demand from the address specified by the SCSI DMA pointer. If the bit is set to 1, data bursts of two words supplied from the 53C94 SCSI controller are written to memory.

SSR<17>

When set to 1, this bit enables SCSI DMA; 0 disables it.

SSR<16>

When set to 1, this bit enables LANCE DMA so that the Ethernet interface can begin data transfer.

SSR<15> DIAGDN

This bit reflects the state of the DIAGDN pin on the motherboard, which is used by manufacturing diagnostics.

SSR<14:13> TXDIS

These bits allow diagnostics to disable the EIA drivers on the serial lines.

When TXDIS are 0's, the EIA drivers are active.
When TXDIS are 1's, the EIA drivers are disabled.

Since the TXDIS signals are automatically cleared at power up or reset, the EIA drivers are enabled by default.

TXDIS<0> disables communication port 1 (serial line 2), and TXDIS<1> disables communication port 2 (serial line 3).

SSR<12>

This bit is reserved.

SSR<11>

This signal can be read from and written to. The SCC UARTS are placed in a hard reset state when this bit is 0. This bit is cleared to 0 at power up or reset, resetting the two SCC's.

SSR<10>

This bit can be read from and written to. The time-of-year controller is placed in a hard reset state when this bit is 0. This bit is cleared to 0 at power up or reset, resetting the TOY. When reset, the TOY loses neither its date nor its 50 bytes of permanent storage.

SSR<9>

This bit can be read from and written to. The 53C94 SCSI controller is placed in a hard reset state when this bit is 0. This bit is cleared to 0 at power up or reset, resetting the 53C94 SCSI controller.

SSR<8>

This bit can be read from and written to. LANCE is placed in a hard reset state when this bit is 0. This bit is cleared to 0 at power up or reset, resetting LANCE.

SSR<7:0>

These bits are reserved; not in use.

C.2.3 System Interrupt Register (SIR)

The SIR register consists of two sections. Bits <31:16> are set by the DMA engine for various DMA conditions. These bits are always set by the system and can be cleared by writing 0 to them. Writing 1 has no effect. These bits are cleared to 0 during system power up or reset.

Bits <15:0> reflect the status of specific system devices and are read-only. A few of these are not usually used as interrupts and should be masked. These bits may or may not be reset to 0 during system power up reset, depending on the state of the interrupting device.

Table C–5: System Interrupt Register 0xBF840110

Bits	Access	Description
31	R/W0C	Communication port 1 transmit page end interrupt
30	R/W0C	Communication port 1 transmit DMA memory read error
29	R/W0C	Communication port 1 receive half page interrupt
28	R/W0C	Communication port 1 receive DMA page overrun
27	R/W0C	Communication port 2 transmit page end interrupt
26	R/W0C	Communication port 2 transmit DMA memory read error
25	R/W0C	Communication port 2 receive half page interrupt
24	R/W0C	Communication port 2 receive DMA overrun
23	R/W0C	Reserved
22	R/W0C	Reserved
21	R/W0C	Reserved
20	R/W0C	Reserved
19	R/W0C	SCSI DMA interrupt, (DMA buffer pointer loaded)
18	R/W0C	SCSI DMA overrun error

Table C–5 (Cont.): System Interrupt Register 0xBF840110

Bits	Access	Description
17	R/W0C	SCSI DMA memory read error
16	R/W0C	LANCE DMA memory read error
15	R	Reserved
14	R	NVR mode jumper
13	R	TURBOchannel slot 2 interrupt
12	R	TURBOchannel slot 1 interrupt
11	R	TURBOchannel slot 0 interrupt
10	R	NRMOD manufacturing mode jumper
9	R	SCSI interrupt from 53C94 SCSI controller
8	R	Ethernet interrupt
7	R	SCC(1) serial interrupt (communication 2)
6	R	SCC(0) serial interrupt (communication 1)
5	R	Reserved
4	R	PSWARN power supply warning indicator
3	R	Reserved
2	R	SCSI data ready
1	R	PBNC
0	R	PBNO

NOTE: *Communication port 1 is the same as serial line 2. Communication port 2 is the same as serial line 3.*

SIR<31>

This interrupt is generated by the communication port 1 transmit DMA logic. The DMA transmitter, when enabled, transmits bytes until the pointer reaches a 4-Kbyte page boundary. At this point, it stops DMA and interrupts the processor. DMA is disabled whenever this bit is set. To restart, clear this bit by writing 0; writing 1 has no effect.

SIR<30>

When a parity error, page crossing error, or maximum transfer length error occurs during a communication transmit port 1 DMA, this bit is set and the

DMA is disabled. The DMA pointer contains the error address. Check the memory sections for more information. To restart, software must clear this bit by writing 0; writing 1 has no effect.

SIR<29>

When the receive DMA pointer associated with communication port 1 reaches a half page (2-Kbyte) boundary, this bit is set. Software must disable DMA and then load a new pointer and restart DMA without being interrupted. Clear this bit by writing 0; writing 1 has no effect. The value of this bit is informational only and does not stop the DMA.

SIR<28>

When the receive DMA pointer associated with communication port 1 reaches a page boundary, this bit is set and the DMA disabled. To restart, clear this bit by writing 0; writing 1 has no effect. Note that bit <29> is set whenever this bit is set.

SIR<27>

This interrupt is generated by the communication port 2 transmit DMA logic. The DMA transmitter, when enabled, transmits bytes until the pointer reaches a page boundary. At this point, it stops DMA and interrupts the processor. DMA is disabled whenever this bit is set. Clear this bit by writing 0; writing 1 has no effect. Clearing this bit may restart the DMA if the DMA enable bit is still on.

SIR<26>

When a parity error, page crossing error, or maximum transfer length error occurs during a communication transmit port 2 DMA, this bit is set and the DMA is disabled. The DMA pointer will contain the error address. Check the memory sections for more information. To restart, software must clear this bit by writing 0; writing 1 has no effect.

SIR<25>

When the receive DMA pointer associated with communication port 2 reaches a (2-Kbyte) half-page boundary, this bit is set. Software must disable DMA, load a new pointer, and restart DMA quickly. Clear this bit by writing 0. Writing 1 has no effect. This bit will always be set when bit 24 is set. The value of this bit is informational only and does not stop the DMA.

SIR<24>

When the receive DMA pointer associated with communication port 2 reaches a page boundary, this bit is set and the DMA disabled. To restart, clear this bit by writing 0; writing 1 has no effect. Note that bit<25> is also set whenever this bit is set.

SIR<23:20>

These bits are reserved.

SIR<19>

This interrupt is set whenever the SCSI DMA buffer pointer associated with the SCSI port is loaded into the SCSI DMA pointer register. Software uses this interrupt to load a new buffer pointer into the SCSI buffer pointer register. Clear this interrupt by writing 0 to it.

SIR<18>

This bit is set when the buffer pointer is not reloaded soon enough. It indicates an overrun condition as the data buffer space is exhausted. DMA is disabled when this bit is set. Clear this bit by writing 0 to it.

SIR<17>

This bit is set when the SCSI DMA encounters a memory read error during a DMA. DMA is disabled when this bit is set. Clear this bit by writing 0 to it.

SIR<16>

This bit is set to 1 when the LANCE DMA encounters a memory read error, disabling DMA. The LANCE will then enter a timeout state, interrupting the processor to handle the problem. The address of the error will be visible in the LPR. Clear this bit by writing 0 to it; writing 1 to it has no effect.

SIR<15>

This bit is reserved.

SIR<14> UNSCUR

When this bit is set, the contents of the NV RAM in the TOY clock chip are set to default system values. Any password that had been saved is lost.

SIR<13> TCO 2 interrupt

This bit reflects the value of the TURBOchannel slot 2 interrupt.

SIR<12> TCO 1 interrupt

This bit reflects the value of the TURBOchannel slot 1 interrupt.

SIR<11> TCO 0 interrupt

This bit reflects the value of the TURBOchannel slot 0 interrupt.

SIR<10> NRMMOD

This bit reflects the state of the manufacturing jumper on the module. When the jumper is absent, NRMMOD is 0, and the console should perform its normal power up or reset tests and boot. When the jumper is installed, NRMMOD is 1, and the console will execute manufacturing tests.

SIR<9>

This bit follows the state of the interrupt from the 53C94 SCSI controller chip. This interrupt indicates that the transfer is complete.

SIR<8>

This bit follows the state of the interrupt from the LANCE.

SIR<7>

This interrupt is generated by SCC(1), which contains the communication port 2 (ch B). Software must read SCC(1) internal registers to determine the appropriate course of action. Communication port 2 is the same as serial line 3.

SIR<6>

This interrupt is generated by SCC(0), which contains both the communication port 1 (ch B). Software must read SCC(0) internal registers to determine the appropriate course of action. Communication port 1 is the same as serial line 2.

SIR<5>

This bit follows the state of the time-of-year clock interrupt.

SIR<4>

This bit follows the state of the power supply warning indicator. When this bit is set, the operating system should report an error. When the power supply overheats, this bit is set to 1.

SIR<3>

This bit is reserved.

SIR<2>

This bit indicates SCSI receive data in the FIFO of the 53C94 SCSI controller. When transfers are aligned and the DMA is enabled, data is moved from the FIFO to main memory by the I/O control ASIC, and this interrupt is masked by software. Unaligned transfers cannot use DMA and thus cannot use this interrupt to signal when the processor must move data to memory.

SIR<1>

This bit reflects the state of the halt button on the back of the system unit. This bit is set to 0 when the button is pushed. This interrupt should always be masked.

SIR<0>

This bit reflects the state of the halt button on the back of the system unit. This bit is set to 1 when the halt button is pushed. On R3000A systems, this interrupt should always be masked. The halt interrupt is also presented at the processor interface, so it should be visible to the CPU.

C.2.4 System Interrupt Mask Register

Table C–6: System Interrupt Mask Register 0xBF840120

Bits	Access	Description
31:0	R/W	Interrupt mask

<31:0>

These bits, if 0, mask the corresponding interrupt observable in the SIR. Bit <0> masks SIR<0>, bit <1> masks SIR<1>, and so on. The mask does not prevent an interrupt from showing up in the SIR; it merely keeps the CPU from being interrupted. All bits of the interrupt mask are set to 0 on power up, masking all interrupts. Software must set to 1 those interrupts that it wants enabled.

C.2.5 Error Address Register (EAR)

The error address register (EAR) (address: 0xBFA40000) is the primary error log register that records the physical address of TC I/O timeouts, TC DMA overruns, and memory ECC errors. The EA register is cleared by system reset or by a processor write. When an error occurs, EA.VALID is set along with the log bits. Table C–7 shows the format of the EA register during reads.

Table C–7: Error Address Register 0xBFA40000

Base	Size	Name
31	1	VALID
30	1	CPU
29	1	WRITE
28	1	ECCERR
27	1	RSRVD
0	27	ADDRESS

EAR<31> - EA.VALID

This bit is set to 1 when error information is clocked into the register. When EA.VALID is already set, error logging is disabled. That is, the EA register indicates only the first error that occurred if there are multiple errors.

EAR<30> - EA.CPU

If this bit is 1, the error occurred during a processor transaction. If this bit is 0, the error occurred during a TC DMA transaction.

EAR<29> - EA.WRITE

If this bit is 1, the error occurred on an I/O write or memory write transaction. If this bit is 0, the error occurred on an I/O read or memory read transaction.

EAR<28> - EA.ECCERR

If this bit is 1, an ECC error occurred. If this bit is 0, an I/O timeout or DMA overrun occurred.

EAR<27> - EA.RSRVD

This bit is reserved and stuck at 0.

EAR<26:0> - EA.ADDRESS

This field records the value of the pipelined address in effect at the time the error occurred. For I/O transactions and partial memory writes, this is the word address issued by the processor. For DMA overrun errors, this is the word address of the last valid word transferred (127). For processor and DMA memory reads, this is the word address in the memory controller. However, due to pipelining of the memory controller, the column field of the word address has advanced five stages before the ECC error status is available. Software must extract ADDRESS[11:0], perform a signed subtract of five, and then reinsert this value into ADDRESS[11:0] to recover the address of the word that contained the ECC error.

Table C–8 lists the values of bits <30>, <29>, and <28> for the different types of system errors. During read conflicts, the memory controller may service the same read request several times (while stalling the processor) until conflicting write data in the write buffer has been flushed. It is possible for ECC read errors to occur during processor reread conflicts when the processor is stalled. However, after the write buffer is flushed, the error is overwritten with new data, so the processor will not receive a bus error on termination of the read. Also, if the processor is waiting for a memory space partial write to complete, and a multi bit ECC error occurs during the read/modify/write of the partial data, invalid data and valid ECC check bits will be loaded into memory. In this case, the ensuing read will complete without causing an exception even though the read data is invalid. If the address is a cached location, invalid data will be loaded into the cache and the cache entry will be incorrectly marked valid. Regardless of the type of masked error, a memory interrupt will be generated, and the offending ECC read error or processor partial write error will be correctly logged in the EA and ES registers.

Table C–8: EA Error Log Types

bit <30> CPU	bit <29> CPU	bit <28> CPU	Error Type
0	0	0	DMA read overrun
0	0	1	DMA memory read
0	1	0	DMA memory write
0	1	1	Invalid combination
1	0	0	Processor I/O read timeout
1	0	1	Processor memory read ECC
1	1	0	Processor I/O write timeout
1	1	1	Processor partial memory write ECC

If the MB ASIC has prefetching enabled, it is possible to log processor read hard ECC errors without a processor error if the ECC error occurs in the prefetched portion of the cache block.

C.2.6 Error Syndrome Register (ES)

The error syndrome (ES) register (address: 0xBFA80000) is a slave error log register that records check bits and syndrome bits of the last memory read. The ES register is frozen when EA <31> is 1. The ES register is cleared by system reset and processor writes.

Table C–9 shows the format of the ES register during reads. The syndrome bytes are only valid if EA <31> is 1. The CHKHI byte is only valid if the VLDHI bit <31> is set to 1. The CHKLO byte is only valid if the VLDLO bit <15> is set to 1.

Table C–9: Error Syndrome Register 0xBFA80000

Base	Size	Name
31	1	VLDHI
24	7	CHKHI
23	1	SNGHI
16	7	SYNHI
15	1	VLDLO
8	7	CHKLO
7	1	SNGLO
0	7	SYNLO

ES<31> - ES.VLDHI

This bit is set to 1 whenever the CHKHI field <30:24> is updated.
ES<30:24> - ES.CHKHI

In the absence of errors, this field records the last check bits read from the high bank of memory (odd word). Once an error occurs, and the EA.VALID bit (EA<31>) is set to 1, this field is frozen.

ES<23> - ES.SNGHI

This bit records the single-versus-double bit error output of the ECC logic at the time that an error was detected by the high bank of memory. If it is 1, a single-bit error occurred. If it is 0, a double-bit error occurred. This bit is valid when the ES.SYNHI <22:16> field is valid.

ES<22:16> - ES.SYNHI

This field records the the syndrome bits calculated by the ECC logic at the time that an error was detected by the high bank of memory (odd words).

The EA.ADDRESS field (EA <26:0>) field must be used to determine whether the error pertains to a low or high word of memory. This field is undefined for low bank errors. The syndrome can be used to determine which bit was in error. See the next section, "ECC logic," for a description of the syndrome logic.

ES<15> - ES.VLDLO

This bit is set to 1 whenever the CHKLO field <14:8> is updated.

ES<14:8> - ES.CHKLO

In the absence of errors, this field records the last check bits read from the low bank of memory (even word). Once an error occurs, and the EA.VALID bit (EA<31>) is set to 1, this field is frozen.

ES<7> - ES.SNGLO

This bit records the single-versus-double bit error output of the ECC logic at the time that an error was detected by the high bank of memory. If it is 1, a single-bit error occurred. If it is 0, a double-bit error occurred. This bit is valid when the ES.SYNLO <6:0> field is valid.

ES<6:0> - ES.SYNLO

This field records the the syndrome bits calculated by the ECC logic at the time that an error was detected by the low bank of memory (even words). The EA.ADDRESS field (EA <26:0>) field must be used to determine whether the error pertains to a low or high word of memory. This field is undefined for high bank errors. The syndrome can be used to determine which bit was in error. See Section C.2.8 for a description of the syndrome logic.

C.2.7 Control Register (CS)

The control (CS) register (address: 0xBFAC0000) controls the memory array size decoding via the CS.BNK32M bit <10>. The CS register also controls the ECC data path. CS is a read/write register that is cleared by system reset.

Table C–10 shows the format of the CS register during reads and writes.

Table C–10: Control Register 0xBFAC0000

Base	Size	Name
16	16	RSRVD2

Table C–10 (Cont.): Control Register 0xBFAC0000

Base	Size	Name
15	1	DIAGCHK
14	1	DIAGGEN
13	1	CORRECT
11	2	RSRVD1
10	1	BNK32M
7	3	RSRVD0
0	7	CHECK

CS<31:16> - CS.RSRVD2

This field must be written with zeros.

CS<15> - CS.DIAGCHK

This bit controls a diagnostic multiplexor in the ECC read data path. the CS.DIAGCHK bit <15> is 0, check bits from the memory array ca are used during memory reads. If the CS DIAGCHK bit <15> is 1, CS.CHECK field <6:0> specifies the check bits during memory reads. Si CS is cleared by system reset, check bits are read from memory by defa

CS<14> - CS.DIAGGEN

This bit controls a diagnostic multiplexor in the ECC write data path. If CS.DIAGGEN bit <14> is 0, check bits are calculated from the processo TC data word during memory writes. If the CS DIAGGEN bit <14> is the CS.CHECK field <6:0> specifies the check bits during memory wri Since CS is cleared by system reset, check bits are generated from proces or TC data by default.

CS<13> - CS.CORRECT

This bit controls whether or not the ECC logic corrects single-bit error memory read data. When this bit is 1, the single-bit error in the read d is complemented as specified by the ECC syndrome. When this bit i and the ECC logic detects a multibit error, the output of the ECC logi undefined. The state of this bit does not affect memory interrupts, e logging, or bus errors; it only controls modification of memory data. Si CS is cleared by system reset, ECC correction is disabled by default.

CS<12:11> - CS.RSRVD1

This field must be written with zeroes.

CS<10> - CS.BNK32M

This bit controls the memory bank stride. If this bit is 0, the stride is 8 Mbytes. If this bit is 1, the stride is 32 Mbytes. Powerup/reset software sets this bit and determines whether each memory module is an 8- or 32-Mbyte module. Then, if no 32-Mbyte modules are found, this bit is cleared. Since CS is cleared by system reset, the memory bank stride defaults to 8 Mbytes.

CS<9:7> - CS.RSRVD0

This field must be written with zeroes.

CS<6:0> - CS.CHECK

This field specifies the diagnostic check value used by the CS.DIAGCHK and CS.DIAGGEN multiplexors.

C.2.8 ECC Logic

This section describes the error correction code (ECC) logic.

MT generates seven check bits for each word written to the memory arrays. For each word read from the memory arrays, MT verifies that the check bits are consistent with the data bits. If a single-bit error is detected, the erroneous bit is automatically corrected if CS CORRECT (bit <>) is 1. If a single- or double-bit error is detected and EA.VALID (bit EA<>) is 0, the EA and ES registers are written and frozen with the address, check, and syndrome bits of the memory word.

Table C–11 lists the data bits included in the exclusive-or logic for each check bit. The ES.CHKLO (ES <>), ES.CHKHI (ES <>), and.CS.CHECK (CS <>) fields correspond to:

64*C16 | 32*C8 | 16*C4 | 8*C1 | 2*C0 | CX

Table C–11: Participating Data Bits in Check Bit Calculation

Bit	Parity	
CX	Even	0 4 6 7 8 9 11 14
C0	Even	0 1 2 4 6 8 10 12
C1	Odd	0 3 4 7 9 10 13 15
C2	Odd	0 1 5 6 7 11 12 13

Table C–11 (Cont.): Participating Data Bits in Check Bit Calculation

Bit	Parity	
C4	Even	2 3 4 5 6 7 14 15
C8	Even	8 9 10 11 12 14 15
C16	Even	0 1 2 3 4 5 6 7
CX	Even	17 18 19 21 26 28 29 31
C0	Even	16 17 18 20 22 24 26 28
C1	Odd	16 19 20 23 25 26 29 31
C2	Odd	16 17 21 22 23 27 28 29
C4	Even	18 19 20 21 22 23 30 31
C8	Even	24 25 26 27 28 29 30 31
C16	Even	24 25 26 27 28 29 30 31

Table C–12 lists the significance of each syndrome code logged in the ES register. The multibit syndrome codes are shown for completeness; MT does not report these as hard errors with the assertion of either p.mc.~rErr or t.mo.~err as appropriate. MT only reports double-bit errors as hard errors. If the operating system detects a multibit error syndrome code, it should log the error and shut down immediately.

Table C–12: Syndrome Decoding

Syndrome	Error	Syndrome	Error	Syndrome	Error	Syndrome	Error
00	None	20	C8	40	C16	60	Double
01	CX	21	Double	41	Double	61	Multi
02	C0	22	Double	42	Double	62	D24
03	Double	23	D8	43	Multi	63	Double
04	C1	24	Double	44	Double	64	D25
05	Double	25	D9	45	Multi	65	Double
06	Double	26	D10	46	Multi	66	Double
07	Multi	27	Double	47	Double	67	D26

Table C–12 (Cont.): Syndrome Decoding

Syndrome	Error	Syndrome	Error	Syndrome	Error	Syndrome	Error
08	C2	28	Double	48	Double	68	D27
09	Double	29	D11	49	Multi	69	Double
0A	Double	2A	D12	4A	D1	6A	Double
0B	D17	2B	Double	4B	Double	6B	D28
0C	Double	2C	D13	4C	Multi	6C	Double
0D	Multi	2D	Double	4D	Double	6D	D29
0E	D16	2E	Double	4E	Double	6E	Multi
0F	Double	2F	Multi	4F	D0	6F	Double
10	C4	30	Double	50	Double	70	D30
11	Double	31	D14	51	Multi	71	Double
12	Double	32	Multi	52	D2	72	Double
13	D18	33	Double	53	Double	73	Multi
14	Double	34	D15	54	D3	74	Double
15	D19	35	Double	55	Double	75	D31
16	D20	36	Double	56	Double	76	Multi
17	Double	37	Multi	57	D4	77	Double
18	Double	38	Multi	58	D5	78	Double
19	D21	39	Double	59	Double	79	Multi
1A	D22	3A	Double	5A	Double	7A	Multi
1B	Double	3B	Multi	5B	D6	7B	Double
1C	D23	3C	Double	5C	Double	7C	Multi
1D	Double	3D	Multi	5D	D7	7D	Double
1E	Double	3E	Multi	5E	Multi	7E	Double
1F	Multi	3F	Double	5F	Double	7F	Multi

Appendix D

Connector Pin Assignments

This appendix lists pin assignments for the following connectors:

- SCSI cable connectors
- Serial communications connectors
- ThickWire Ethernet connectors
- Modem loopback connectors
- Ethernet loopback connectors

It also provides a summary of loopback connectors

Table D-1: SCSI Cable Connector Pin Assignments

Pin	Signal	Pin	Signal
50	~ I/O	25	GND
49	~ REQ	24	GND
48	~ C/O	23	GND
47	~ SEL	22	GND
46	~ MSG	21	GND
45	~ RST	20	GND
44	~ ACK	19	GND
43	~ BSY	18	GND
42	GND	17	GND
41	~ ATN	16	GND
40	GND	15	GND
39	RSVD	14	GND
38	TERMPWR	13	NC
37	RSVD	12	GND
36	GND	11	GND
35	GND	10	GND
34	~ PARITY	9	GND
33	~ DATA<7>	8	GND
32	~ DATA<6>	7	GND
31	~ DATA<5>	6	GND
30	~ DATA<4>	5	GND
29	~ DATA<3>	4	GND
28	~ DATA<2>	3	GND
27	~ DATA<1>	2	GND
26	~ DATA<0>	1	GND

Table D–2: Serial Communications Connectors Pin Assignments

Pin	Source	Signal	CCITT[1]	EIA[2]	Description
1		GND	102	AB	Signal ground
2	KNO3A-AA	TX	103	BA	Modem transmitted data
3	Modem/printer	RX	104	BB	Modem received data
4	KNO3A-AA	RTS	105	CA	Request to send
5	Modem/printer	CTS	106	CB	Clear to send
6	Modem/printer	DSR	107	CC	Data set ready
7		GND	102	AB	Signal ground
8	Modem/printer	CD	109	CF	Carrier detector
9					Unconnected
10					Unconnected
11					Unconnected
12	Modem/printer	SI	112	CI	SPDMI
13					Unconnected
14					Unconnected
15	Modem/printer	TxCk (DCE)	114	DB	Modem transmit clock
16					Unconnected
17	Modem/printer	RxCk (DEC)	115	DD	Modem transmit clock
18					Unconnected
19					Unconnected
20	KNO3A-AA	DTR	108.2	CD	Data terminal ready
21					Unconnected
22	Modem/printer	RI	125	CE	Ring indicator
23	KNO3A-AA	SS	111	CH	DSRS
24					Unconnected
25					Unconnected

[1]Comite Consultatif International Telegraphique et Telephonique, an international consultative committee that sets international communications standards
[2]Electronic Industries Association

Table D–3: ThickWire Ethernet Connector Pin Assignments

Pin	Source	Signal	Description
1			Shield
2	XCVR	ACOL+	Collision presence
3	KNO3A-AA	ATX+	Transmission
4		GND	Ground
5	XCVR	ARX+	Reception
6	XCVR	GND	Power return
7		CTL+	Control output
8		GND	Ground
9	XCVR	ACOL–	Collision presence
10	KNO3A-AA	ATX–	Transmission
11		GND	Ground
12	XCVR	ARX–	Reception
13	KNO3A-AA	+12V	Power
14		GND	Ground
15		CTL–	Control output

Table D–4: Power Supply Pin Assignments

Cable	Pin	Signal
Red		5 volt supply
Black		5 volt return
Multilead	1	POK
	2	+12 volt return
	3	+12 volt supply
	4	WARN
	5	–12 volt return
	6	–12 volt supply

Table D–5: Modem Loopback Connector Pin Assignments

From Pin No.	Signal	To Pin No.	Signal
P4-2	TX2	P4-3	RX2
P4-4	RTS2	P4-5	CTS2
P4-6	DSR2	P4-20	DTR2
P4-12	SPDMI2	P4-23	DSRS2
P4-18	LLPBK2	P4-8	CI2
P4-18	LLPBK2	P4-22	RI2
P4-18	LLPBK2	P4-25	TMI2

Table D–6: Ethernet Loopback Connector Pin Assignments

From Pin No.	Signal	To Pin No.	Signal	Description
P6-3	TRA+	P6-5	REC+	Through capacitor
P6-10	TRA–	P6-12	REC–	Through capacitor
P6-13	PWR	P6-6	RET	Through resistor and LED

Table D–7: Summary of Loopback Connectors

Function	Standard/ Unique	Part Number	Option Number
Communications loopback connector	Standard	12-15336-13	H3200
ThickWire loopback connector	Standard	12-22196-02	N/A
ThinWire T-connector	Standard	12-25869-01	H8223
ThinWire terminator	Standard	12-26318-01	H8225

Appendix E
ULTRIX System Exercisers

The ULTRIX operating system contains a set of commands called exercisers. The exercises reside in the **/usr/field** directory and allow you to test all or part of your system by exercising specified parts.

NOTE: *The ULTRIX exercisers are not a mandatory subset and may not be installed on your system. Subset UDTEXER must be installed for the exercisers to be present.*
The following ULTRIX-based exercisers are currently available and can be used to exercise and test the DECsystem 5900;

- fsx = file system exerciser
- memx = memory exerciser
- shmx = shared memory exerciser
- dskx = disk exerciser
- mtx = magnetic tape exerciser
- tapex = tape exerciser program
- netx = tcp/ip network exerciser
- cmx = communications exerciser
- lpx = line printer exerciser

To run these exercisers, the operator must be logged in as superuser (root) and then change directory to /usr/field.

All of the exercisers can be run in either the foreground or the background and can be canceled at any time by pressing CTRL/C in the foreground. More than one exerciser can be run at the same time.

To run more than one exerciser simultaneously, a shell script called syscript is used. The syscript command asks that which exercisers are to be run, how long the exercisers will be run and how many exercisers are to be run at one time. The syscript command can be used to exercise a device, a subsystem, or the entire system.

Each time an exerciser is invoked, a new logfile is generated in the /usr/field directory. The logfile is record of the exerciser's results and consists of the starting and stopping times, and of error and statistical information.

E.1 File system Exerciser (fsx)

The file system exerciser (fsx) is used to exercise a file system locally. Fsx exercises the specified file system by initiating multiple processes which creates, writes, closes, opens, reads, validates, and unlinks a test file of random data. The format of the fsx command is:

```
fsx -h -ofile -pn -fpath -tmin &

-h       Prints the help message for fsx.
-ofile   Saves the output in the designated file.
-pn      Specifies the number of process to initiate. The
  maximum is 250, the default is 20.
-fpath   Specifies pathname of the directory of the file
  system to test. The default is /usr/field.
-tmin    Specifies the number minutes fsx is to run
&        Runs fsx in the background.
```

The following example starts 5 processes and tests the /usr file system for 60 minutes in the background:

```
# fsx -p5 -f/usr -t60 &
```

E.2 Memory Exerciser (memx)

The memx command exercises system memory. The memx command runs ones and zeros, zeros and ones and random data patterns in the allocated memory being tested. The format of the memx commands is:

```
memx -h -ofile -s -mn -px -tmin &

-h      Prints the help message for the memx command.
-ofile  Saves the output in the designated file.
-s      Disables automatic invocation of shmx.
-mn     Specifies the ammount of memory in bytes to test.
 The default is total memory divided by 20.
-px     Specifies the number of process to initiate.
 The maximum is 20 which is also the default.
-tmin   Specifies the number of minutes memx is to run.
&       Runs memx in the background.
```

The following example disables the shared memory exerciser, tests 4095 bytes of memory, starts 5 processes and run for 60 minutes in the background.

```
# memx -s -m4095 -p5 -t60 &
```

E.3 Shared Memory Exerciser (shmx)

The shmx command tests shared memory. Shmx spawns a background process called shmxb, and together shmx and shmxb exercise the shared memory segments. They take turns writing and reading each other's data. The format of shmx is:

```
shmx -h -ofile -ti -mj -sk -v &

-h       Prints help message for the shmx command.
-ofile   Saves output in designated file.
-ti      Indicates the run time in mimutes (i).
-mj      Specifies the memory segment size in bytes (j)
  to be tested.
-sk      Specifies the number if memory segments (k).
  The maximum and default is 6.
-v       Uses the fork system call instead of the vfork
  system call to spawn shmxb.
&        Runs shmx in the background.
```

The following example runs shmx for 180 minutes, tests 100,000 bytes on three memory segments and runs in the background.

```
# shmx -t180 -m100000 -s3 &
```

E.4 Disk Exerciser (dskx)

The dskx command exercises disk drives. The dskx command exercises specified partitions and file systems on the designated disk. The format of the dskx command is:

CAUTION: *The -p and -c options destroy data on the device you are testing. Use extreme caution when using either of this options.*

```
dskx -h -ofile -pdevpart -cdev -rdev -tmin -dn &

-h          Prints the help message for the dskx command.
-ofile      Saves the output in designated file.
-pdevpart   Performs random seeks, writes and reads on the
     specified partition (part) of the device (dev).
            Next, it validates the random data and block sizes.
            You cannot test the c partition, because the test
            would corrupt the bad block information.
-cdev       Performs random seeks, writes and reads on all
     partitions of the specified device (dev), except
            the c partition. Partition c is not tested because
            the test would corrupt bad block information.
-rdev       Performs random seeks and reads on all partitions
     of the device (dev), except partition c. The -r
            option is safe to use on all disks because it will
```

```
            not overwrite data.
-tmin       Specifies the run time in minutes.
-dmin       Specifies in minutes how often you want the dskx
     command to print diagnostics to the terminal. The
            default is to print diagnostics upon completion of
            the exercise.
&           Runs dskx in the background.
```

The following example runs dskx on rz0 for 20 minutes and diagnostic information is displayed on the terminal every 5 minutes.

```
# dskx -rrz0 -t20 -d5
```

E.5 Mag Tape Exerciser (mtx)

The mtx command writes, reads and validates random data on a tape device from BOT to EOT. The format of the mtx command is:

```
mtx -h -ofile -rn -fn -sdev# -ldev# -vdev# -adev# -tmin &
```

```
-h        Prints help message for the mtx command.
-ofile    Saves output in designated file.
-rn       Specifies record length in bytes, default is 10240.
-fn       Specifies the length of the files in numbers of records.
-sdev#    Performs a short record test that writes, reads and
  validates 512-byte records on device (dev). The dev#
  variable is the raw device name and number, such as
  rmt0h.
-ldev#    Performs a long record test that writes, reads and
  validates 10240-byte records on device (dev). The dev#
  variable is the raw device name and number, such as
  rmt0h.
-vdev#    Performs a variable record length test that writes,
  reads and validates random record lengths from 512 to
  20280 bytes on dev (dev). The dev# variable is the raw
  device name and number, such as rmt0h.
-adev#    Performs short, long and variable record length tests on
  device (dev). The dev# variable is the raw device name
  and number, such as rmt0h.
-tmin     Specifies the run time in minutes.
&         Runs mtx in the background.
```

The following example writes 20480 byte records to rmt0h for 60 minutes and runs in the background:

```
# mtx -r20480 -lrmt0h -t60 &
```

E.6 Tape Exerciser (tapex)

The tapex command is similar to the mtx command but performs additional tests, for example, positioning tests for records and files. The are over 30 options that can be used for the tapex command and space does not allow for their inclusion here. To view the tapex command help file use the following command:

```
# tapex -h
```

E.7 Network Exerciser (netx)

The netx command exercises the TCP/IP network. The netx command sets up a stream socket connection with netx acting as the client and the miscd utility acting as the server in the TCP/IP internet domain. Using the connection, netx writes random data to the miscd server. The server loops the data back to netx, and then the data is read and verified against the original data. The format of the netx command is:

```
netx -h -pn nodename -tmin &

-h            Prints the help message for netx.
-pn           Specifies the port number to use in the internet
domain. The variable n must be less than 32768.
nodename      The name of the remote or local system host
running the server.
-tmin         The run time in minutes
&             Runs netx in the background.
```

The following example runs a test on node tinker for 60 minutes in the background.

```
# netc tinker -t60 &
```

E.8 Communications Exerciser (cmx)

The cmx command exercises terminal communications. The cmx command writes, reads and validates random data and packet lengths on the communications line or lines specified. The format of the cmx command is:

```
cmx -h -ofile -tmin -l line-1 line-2 line-n... &
```

```
-h         Prints the help message of cmx.
-ofile     Saves the output to the designated file.
-tmin      Specifies the run time in minutes.
-l line..  Specifies the line you want to test.
&          Runs cmx in the background.
```

The following example runs a test on tty00 for 45 minutes in the background.

```
# cmx -l 00 -t45 &
```

E.9 Line Printer Exerciser (lpx)

The lpx command exercises line printers. The lpx command exercises line printers by printing a rolling character pattern repeatedly to the device. The format of the lpx command is:

```
lpx -h -ofile -pn -ddev -tmin &
```

```
-h       Prints the help message for the lpx command.
-ofile   Saves the output to the designated file
-pn      Specifies the pause period in n minutes.
-ddev    Specifies the line printer you want to test.
-tmin    Specifies the run time in minutes.
&        Runs lpx in the background.
```

The following example runs a test on lp1 for 60 minutes in the background,

```
# lpx -t60 -dlp1 &
```

Index

U