SunVTS™ 6.1 Test Reference Manual for x86 Platforms

Sun Microsystems, Inc.
www.sun.com

Part No. 819-2373-10
January 2006, Revision A

Submit comments about this document at: http://www.sun.com/hwdocs/feedback

Please
Recycle

Contents

Preface

SunVTS™ is the Sun Microsystems™ Validation Test Suite. SunVTS is a comprehensive software diagnostic package that tests and validates Sun SPARC and x86 hardware by verifying the configuration and functionality of controllers, devices, and platforms.

SunVTS is primarily used from a graphical user interface (GUI), for the Common Desktop Environment (CDE). This book describes SunVTS tests that run on machines with SPARC™ and x86 architectures. The descriptions include specific test options, procedures, and error messages.

This book is primarily written as a reference for SunVTS test specific information. Refer to the *SunVTS User's Guide* for overall SunVTS information. Developers or experienced users who want to run the SunVTS diagnostic application will find these documents useful.

Note – The Solaris release with which this version of SunVTS is delivered supports systems that use the SPARC® and x86 families of processor architectures: UltraSPARC®, SPARC64, AMD64, Pentium, and Xeon EM64T. The supported systems appear in the *Solaris 10 Hardware Compatibility List* at `http://www.sun.com/bigadmin/hcl`. This document cites any implementation differences between the platform types.

In this document the term x86 refers to 64-bit and 32-bit systems manufactured using processors compatible with the AMD64 or Intel Xeon/Pentium product families. For supported systems, see the *Solaris 10 Hardware Compatibility List*.

Before You Read This Book

In order to fully use the information in this document, you must have thorough knowledge of the topics discussed in these books:

- *SunVTS User's Guide*
- *SunVTS Quick Reference Card*

How This Book Is Organized

This book is organized as follows:

Chapter 1 describes SunVTS requirements, test modes, user interfaces, the collection of tests, and how to run a test from the command line.

The remaining chapters describe the individual SunVTS tests, their options, applicable test modes, and command-line syntax. These chapters are arranged in alphabetical order according to each test name.

Appendix A provides information about the serial and parallel port loopback connectors that are required by some of the SunVTS tests.

Using UNIX Commands

This document might not contain information on basic UNIX® commands and procedures such as shutting down the system, booting the system, and configuring devices.

Refer to one or more of the following for this information:

- *Solaris Handbook for Sun Peripherals*
- Software documentation that you received with your system
- Solaris™ Operating System documentation, which is at

  `http://docs.sun.com`

Shell Prompts

Shell	Prompt
C shell	*machine-name*%
C shell superuser	*machine-name*#
Bourne shell and Korn shell	$
Bourne shell and Korn shell superuser	#

Typographic Conventions

Typeface or Symbol	Meaning	Examples
AaBbCc123	The names of commands, files, and directories; on-screen computer output	Edit your .login file. Use ls -a to list all files. % You have mail.
AaBbCc123	What you type, when contrasted with on-screen computer output	% **su** Password:
AaBbCc123	Book titles, new words or terms, words to be emphasized. Replace command-line variables with real names or values.	Read Chapter 6 in the *User's Guide*. These are called *class* options. To delete a file, type **rm** *filename*.

Related Documentation

This manual covers SunVTS tests. It serves as a reference companion to the SunVTS documents listed below.

Application	Title	Part Number
Installation and Navigation	*SunVTS 6.1 User's Guide*	819-2361-10
Quick Reference Card	*SunVTS Quick Reference Card*	819-2365-10

Accessing Sun Documentation

You can view, print, or purchase a broad selection of Sun documentation, including localized versions, at:

http://www.sun.com/documentation

Third-Party Web Sites

Sun is not responsible for the availability of third-party web sites mentioned in this document. Sun does not endorse and is not responsible or liable for any content, advertising, products, or other materials that are available on or through such sites or resources. Sun will not be responsible or liable for any actual or alleged damage or loss caused by or in connection with the use of or reliance on any such content, goods, or services that are available on or through such sites or resources.

Contacting Sun Technical Support

If you have technical questions about this product that are not answered in this document, go to:

http://www.sun.com/service/contacting

Sun Welcomes Your Comments

Sun is interested in improving its documentation and welcomes your comments and suggestions. You can submit your comments by going to:

http://www.sun.com/hwdocs/feedback

Please include the title and part number of your document with your feedback:

SunVTS 6.1 Test Reference Manual for x86 Platforms, part number 819-2373-10

CHAPTER 1

Introduction

This manual describes SunVTS Version 6.1 tests that are distributed on the Solaris 10 1/06 (Solaris 10 Update 1) Software CDs.

This chapter contains the following topics:

The Sun™ Validation and Test Suite (SunVTS) software performs multiple diagnostic hardware tests from a single user interface. SunVTS verifies the connectivity, functionality, and reliability of controllers and devices.

Use SunVTS to test one device or multiple devices. Some of the major test categories are as follows:

- Audio tests
- Communication (serial and parallel) tests
- Graphic/video tests
- Memory tests
- Network tests
- Peripherals (disks, tape, CD-ROM, DVD-ROM, printer, diskettes) tests
- Processor tests
- Storage tests

SunVTS is comprised of many individual tests that support testing of a wide range of products and peripherals. Most of the tests are capable of testing devices in a 32-bit or 64-bit Solaris environment.

Such flexibility means that the proper test modes and options need to be selected to maximize its effectiveness. This book covers the individual test options, modes, and requirements. For overall test configuration modes and options refer to the *SunVTS User's Guide*.

Note – When an error occurs in VTS testing, the test message window displays the error number, the error description, the probable cause of the error, and the recommended actions. Because this information is displayed at the time of the error, error messages are not included in this manual.

The default installation directory for SunVTS is `/opt/SUNWvts`. However, when you are installing SunVTS, you can specify a different directory. Refer to the *SunVTS User's Guide* for installation information.

Note – SunVTS does not support processor sets. If processor sets are defined, you must first delete the processor sets before running SunVTS.

SunVTS Version Information

The standard command line argument, `-V`, displays the SunVTS version and release date of the test if available.

SunPCi-3 Card Support

`sunpci2test` now supports the SunPCi-3 cards. Solaris 10 supports SunPCi-3 Version 3.2.2 with Patch 118591-01 only. Solaris 10 does not support the SunPCi-2 card.

x86 and 64-Bit Solaris Support

Starting with Solaris 10, the SunVTS infrastructure and core diagnostics are available for x86 Solaris platforms. Starting with Solaris 10 7/05, SunVTS x86 diagnostics are supported in the AMD 64-bit environment for the SunVTS kernel (`vtsk`). All x86 diagnostics except the System Test (`systest`) are ported to 64-bit.

x86 support is added for the Level 1 Data Cache Test (`l1dcachetest`) in this release.

SunVTS is supported and tested on the following Sun x86 platforms:

- Sun Fire V60x
- Sun LX50
- Sun Fire B100x

- Sun Fire B200x
- Sun Fire V20z
- Sun Fire V40z

You must install the x86 version of the SunVTS packages to perform SunVTS on x86 platforms. The software packages use the same names as in the SPARC environment. The SunVTS packages delivered separately for both x86 and SPARC Solaris platforms are as follows:

- `SUNWvts` – Contains the SunVTS core framework that includes the kernel and user interface.
- `SUNWvtsmn` – Contains the SunVTS online manual pages
- `SUNWvtsr` – Contains the SunVTS framework configuration files in the root partition (Superuser).
- `SUNWvtsts` – Contains the SunVTS test binaries.

The SunVTS components available for x86 Solaris platforms are as follows.

Infrastructure:

- `sunvts`
- `vtsk`
- `vts_cmd`
- `vtstty`
- `vtsui`
- `vtsprobe`

SunVTS Tests:

- CD DVD Test (`cddvdtest`)
- CPU Test (`cputest`)
- Disk and Floppy Drives Test (`disktest`)
- Data Translation Look-aside Buffer (`dtlbtest`)
- Floating Point Unit Test (`fputest`)
- InfiniBand Host Channel Adapter (`ibhcatest`)
- Level 1 Data Cache Test (`l1dcachetest`)
- Ethernet Loopback Test (`netlbtest`)
- Network Hardware Test (`nettest`)
- Physical Memory Test (`pmemtest`)
- RAM test (`ramtest`)
- Serial Port Test (`serialtest`)
- System Test (`systest`)
- Universal Serial Board Test (`usbtest`)
- Virtual Memory Test (`vmemtest`)

Test Requirements

SunVTS Version 6.1 was first introduced and designed to run in the Solaris 10 1/06 (Solaris 10 Update 1) Operating System (OS) and subsequent compatible releases. SunVTS 6.1 is not supported on OS releases prior to Solaris 10 3/05 (Solaris 10).

The operating system kernel must be configured to support all peripherals that are to be tested.

Some SunVTS tests have special requirements such as the connection of loopback connectors, installation of test media, or the availability of disk space. These requirements are listed for each test in the corresponding chapter in this book.

Collection of SunVTS Tests

Many individual tests make up the collection of tests in the SunVTS application. Each test is a separate process from the SunVTS kernel. Each test can be run individually from the command line or from the SunVTS user interface.

When SunVTS is started, the SunVTS kernel automatically probes the system kernel to determine the hardware devices. The devices are then displayed on the SunVTS control panel with the appropriate tests and test options. This provides a quick check of your hardware configuration, and no time is wasted trying to run tests that are not applicable to your configuration.

During testing, the hardware tests send the test status and messages to the SunVTS kernel through interprocess communication (IPC) protocols. The kernel passes the status to the user interface and logs the messages.

SunVTS has a shared object library that contains test-specific probing routines. At runtime, the SunVTS kernel dynamically links in and calls these probing routines to initialize its data structure with test-specific information. You can add new tests into the SunVTS environment without recompiling the SunVTS source code.

As of SunVTS 3.0, the SunVTS kernel and most tests support 32-bit and 64-bit operating systems. When the `sunvts` command is used to start SunVTS, the appropriate tests (32-bit or 64-bit versions) are presented.

32-Bit and 64-Bit Tests

Because each test is a separate program, you can run individual tests directly from the command line. When this is done, care must be taken to run the appropriate test (32-bit or 64-bit) that corresponds to the operating system that is running (32-bit or 64-bit). This is done by running tests from specific directories as follows:

- 32-bit tests—`/opt/SUNWvts/bin/`*testname*
- 64-bit tests—`/opt/SUNWvts/bin/64/`*testname*
  - The test is an actual 64-bit binary test if *testname* is a binary file.
  - The test is a 32-bit test capable of running in the 64-bit environment if *testname* is a symbolic link.

Note – The `SUNWvtsx` package must be installed for 64-bit SunVTS support. For more information on SunVTS packages and installation procedures refer to the *SunVTS User's Guide*.

If you use the `sunvts` command to run SunVTS, SunVTS automatically allocates 32-bit or 64-bit tests based on the 32-bit or 64-bit Solaris operating environment that is running. Therefore, the only time that you need to be concerned with the 32-bit or 64-bit operation is when you run the SunVTS kernel or SunVTS tests from the command line.

If you are not sure which operating system is running, refer to the Solaris System Administration manuals. In Solaris 10, you can use the following command to identify the application support of your system.

```
# isainfo -v
```

Note – The `isainfo` command is not available in Solaris 2.6 or earlier releases.

SunVTS User Interfaces

You can run SunVTS tests from various interfaces: The CDE graphical user interfaces or the TTY interface. SunVTS tests can also be run individually from a shell tool command line, using the command-line syntax for each test (refer to "Running a Test From the Command Line" on page 8). TABLE 1-1 describes the various SunVTS user interfaces. Refer to the *SunVTS User's Guide* for more information on these interfaces.

TABLE 1-1 SunVTS System Interfaces

SunVTS System Interfaces	Description
Graphical user interfaces (GUIs)	Users can select tests and test options by pointing and clicking with a mouse button in the CDE interface.
TTY interface	Users can run SunVTS from a terminal or modem attached to a serial port. This feature requires that users use the keyboard instead of the mouse, and it displays one screen of information at a time.
Command-line execution	Users can run each of the SunVTS tests individually from a shell tool command line using the command-line syntax. Each test description in this book contains the corresponding command-line syntax.

Note – To increase or decrease a numeric value in a SunVTS CDE dialog box, you can use either the up or down arrows, or type a new value in the text box and press Return. Select Apply to apply all dialog box changes.

Running a Test From a User Interface

The common way to run SunVTS testing is through a SunVTS user interface—CDE or the TTY interface.

Test configuration, control, and results are easily accessed through buttons and dialog boxes. These buttons and dialog boxes are covered in the *SunVTS User's Guide*. However, the Test Parameter Options dialog box is unique for each test, and is therefore covered in this manual.

Test Parameter Options Dialog Box

The options displayed in this menu differ for each test, but the lower set of buttons are generic and are described below.

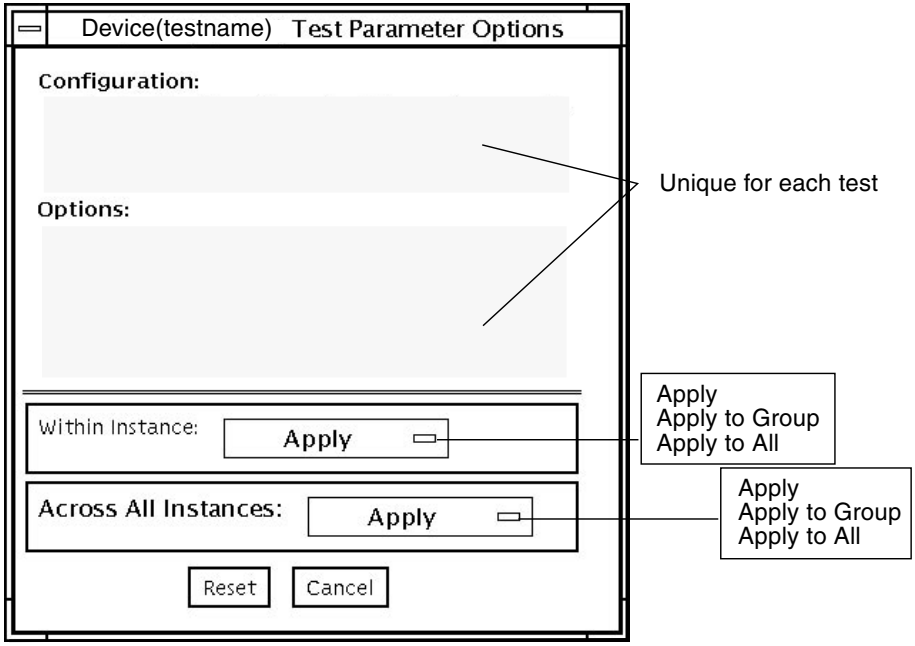


FIGURE 1-1 Test Parameter Options Dialog Box (CDE)

TABLE 1-2 Test Parameter Options Dialog Box Items

Menu Item	Description
Configuration	Information such as device type, capacity, revision, and serial numbers for the selected device. This information cannot be changed.
Options	A list of test options that are used to customize the testing of the selectable device, group, or all devices. The options are specific for each test and are covered in the test specific-chapters in this manual.
Within Instance	Provides the means to apply the settings: • to this device only with Apply, or • to all devices within this group with Apply to Group, or • to all devices (of the *same device type* for *all controllers*) with Apply to All. The option settings are only applied to one instance of the test.

TABLE 1-2 Test Parameter Options Dialog Box Items *(Continued)*

Menu Item	Description
Across All Instances	Provides the means to apply the settings globally: • to this device only with Apply, or • to all devices within this group with Apply to Group, or • to all devices (of the *same device type* for *all controllers*) with Apply to All. The option settings are applied to all instances.
Reset	Returns the option values to their default settings and closes the test parameter option menu.
Cancel	Ignores any changes made to option values and closes the test parameter option menu.

Note – The Test Parameter Options Dialog Box descriptions also apply to the Test Parameter Options menu in the TTY interface.

Running a Test From the Command Line

In some cases it may be more convenient to run a single SunVTS test from the command line rather than through a SunVTS user interface. The following information describes how to do this.

Unless specified, the test runs without the SunVTS kernel (`vtsk`). All events and errors are sent to `stdout` or `stderr` and are not logged in the log files.

When you run a test in this way, you must specify all test options in the form of command-line arguments.

There are two types of command-line arguments:

- Standard arguments—common to all tests. Refer to TABLE 1-3 for details.
- Test specific arguments—unique to a specific test. Refer to the test-specific chapters in this book for details.

The standard syntax for all SunVTS tests is:

`testname` `[`**`-scruvdtelnf`**`]` `[`**`-i`** *number*`]` `[`**`-w`** *number*`]` `[`**`-o`** *test_specific_arguments*`]`

Note – 64-bit tests are located in the `sparcv9` subdirectory: `/opt/SUNWvts/bin/sparcv9/`*testname*, or the relative path to which you installed SunVTS. If a test is not present in this directory, then it might be available as a 32-bit test only. For more information, see "32-Bit and 64-Bit Tests" on page 5.

Standard Command-Line Arguments

The following table defines the standard SunVTS command-line arguments:

TABLE 1-3 Standard Command-Line Arguments

Argument	Description
`-s`	Runs a test as though it were invoked from the SunVTS kernel (`vtsk`). The default is to send the output to `stdout` or `stderr`.
`-c`	Enables a core image of the test process to be created in the current working directory upon receipt of certain signals, otherwise those signals are caught and handled to prevent a core from being generated. The default is to disable the creation of a core image.
`-r`	Enables run on error so that when an error occurs, the test continues with the next test sequence instead of exiting. The default is false.
`-v`	Runs the test in Verbose mode and displays messages with more detailed information about the testing process. The default is false.
`-V`	Displays the SunVTS version and release date of the test.
`-d`	Runs the test in debug mode and displays messages to help programmers debug their test code. The default is false.
`-t`	Runs the test in Trace mode and displays messages that track function calls and sequences currently in use by the test code. The default is false.
`-l`	Runs the test in Online Functional mode. This is the same mode that tests run in when executed with the `vtsui.online` command. It is a non-intrusive version that will not significantly affect other applications. See the note below. The default is true.
`-x`	Runs the test in Exclusive mode.
`-n`	Runs the test in Connection mode. See the note below. The default is false.
`-f`	Runs the test in full Functional mode. This mode assumes that the test has complete control of the device under test. See the note below. The default is false.
`-p` *number*	Defines the number of passes for scalable tests. The default is 1.

TABLE 1-3 Standard Command-Line Arguments *(Continued)*

Argument	Description
`-i` *number*	Defines the number of instances for scalable tests. The default is 1.
`-w` *number*	Defines to which instance the test is assigned; this option is for scalable tests. The default is 0.
`-o`	Indicates that the options and arguments that follow are test specific.

Note – Separate each test-specific argument by commas, with no space after each comma.

Note – If you choose to specify a test mode with the `l`, `n`, or `f` option, specify only one option at a time because only one test mode can be selected at a time.

Test-Specific Arguments

There are test-specific arguments, as described in TABLE 1-4. Test-specific arguments follow the format specified in the `getsubopt(3c)` man page. For information about test-specific arguments refer to the specific test chapter in this book.

TABLE 1-4 SunVTS Test-Specific Arguments

Argument	Description
`-o`	Separate each test-specific argument by commas, with no space after the comma. For example: `#./sample -v -o dev=/dev/audio,volume=78` The test option format is specified by the man page `getsubopt`(3C).

Testing Frame Buffers

Before running a frame buffer test, determine whether the test requires frame buffer locking. Not all frame buffer tests have a locking option. Some tests set the lock automatically. Check the test chapter for each individual test to see if this step is needed. If locking is required, you can set the lock in one of two ways:

- If you are using the CDE SunVTS interface, go to the Option menu of the graphic test and select Enable for the frame buffer locking option.
- If you are working from the command line, you can enable frame buffer locking with the `lock=e/d` option. For example, to run the generic frame buffer test (`fbtest`) with a locked frame buffer, type the following:

```
# ./fbtest -o dev=cgthree0,lock=enable
```

(See the test command line argument descriptions in this manual.)

Caution – If frame buffer locking is disabled (unlocked) on frame buffers that are running `vtsui`, or if you move the mouse, you will receive false error messages. Even a slight mouse movement can cause a test to fail.

Caution – Disable the Power Management screen saver option and the Save/Resume option before you run any of the SunVTS frame buffer tests. For information on disabling these Power Management features, refer to the Power Management chapter in the *Solaris Common Desktop Environment: Users's Guide* in the Solaris 9 User Collection. This document is available at: `docs.sun.com`.

Caution – If you are using the CDE interface for SunVTS, do not conduct frame buffer tests through the `dtlogin` window. Log in as `root` and disable the auto-logout option.

Caution – Do not run TTY mode and frame buffer tests concurrently on the console monitor. The frame buffer test may fail.

Testing Multiple Frame Buffers

The following rules apply when you test multiple frame buffers (displays) simultaneously:

- Only the console monitor can run the window environment (such as CDE). The console monitor is the monitor connected to the frame buffer appointed by `/dev/fb`. SunVTS enables frame buffer locking on the console monitor by default.

- The frame buffer that is running the window environment must have window locking enabled to avoid false test failures. All other frame buffers must have window locking disabled.

Remote Testing of Frame Buffers

If you start `sunvts` or `vtsk` from a screen other than the console monitor, frame buffer locking is not available. In this case:

- Disable the window locking option on the remote screen by setting it to `d`.
- Enable frame buffer locking for the console monitor, as shown in the example above. The SunVTS user interface cannot display on a monitor if locking is disabled.

Do not run any graphic programs (including `vtsui`) on the remote frame buffer during graphic testing.

CHAPTER 2

BMC Environment Test (bmcenvironment)

The bmcenvironment test scans the sensors of the Base Management Controller (BMC) and reports failures on sensors that report readings outside of their reported ranges.

Note – The bmcenvironment test is supported on x86 platforms only.

Caution – Ensure that no other applications, such as ipmitool, are using the BMC when perfoming the bmcenvironment test.

Caution – Ensure that a service processor daughter card is installed and seated correctly.

bmcenvironment Options

To reach the following dialog box, right-click on the test name in the System Map and select Test Parameter Options. If you do not see this test in the System Map, you might need to expand the collapsed groups, or your system might not include the device appropriate to this test. Refer to the *SunVTS User's Guide* for more details.

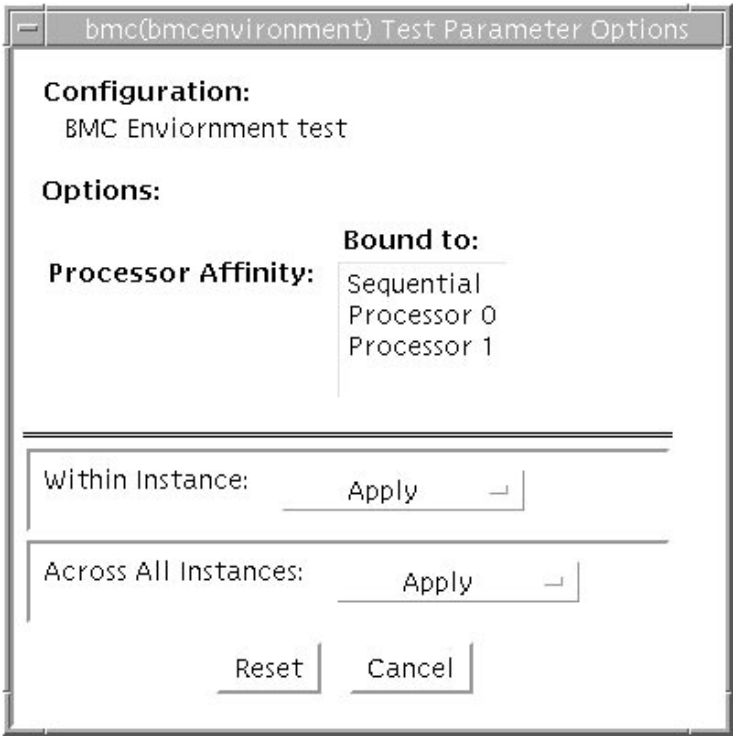


FIGURE 2-1 `bmcenvironment` Test Parameter Options Dialog Box

Note – Only the standard options are supported for the `bmcenvironment` test.

`bmcenvironment` Test Modes

`bmcenvironment` supports the test modes listed in the following table.

TABLE 2-1 `bmcenvironment` Supported Test Modes

Test Mode	Description
Connection	Runs the entire test.

TABLE 2-1 `bmcenvironment` Supported Test Modes

Test Mode	Description
Functional	Runs the entire test.
Online	Runs the entire test.
Exclusive	Runs the entire test.

`bmcenvironment` Command-Line Syntax

`/opt/SUNWvts/bin/bmcenvironment` *standard_arguments*

Note – Only the standard SunVTS command line argumentst are supported for the `bmcenvironment` test.

CHAPTER 3

Optical Disk Drive Test (cddvdtest)

cddvdtest Description

cddvdtest verifies the functionality of optical disk drives.The disktest probe detects the media type in the drive and shows the test options for the media found. If inserted media is not supported by the drive, the disktest probe shows an error and registers options for CD-ROM as default.

Note – cddvdtest is a newly consolidated test which is used to test the optical media drives such as CD-ROM, DVD-ROM, and CD-DVD-RW drives. cddvdtest supports the same set of options for both SPARC and x86.

Volume Management

cddvdtest tests the optical drive(s) even if the volume manager is not running. If the volume manager is running and no media is installed in the drive(s), SunVTS prompts you to install media in the drive before selecting the test. The test fails if you try to run it without media in the drive.

Note – When testing rewritable media, the media can be blank or can contain the SunVTS test data. When testing write-once media, the media (such as CD-R) must be blank at the start the write test. Such media could still run multiple passes of the test, because after the first write test, the subsequent tests treat the media as read only and perform the test accordingly.

For CD-ROM and DVD-ROM drives, the test checks the unit by reading either the CD or DVD. For CD-ROMs, each track is classified as follows:

- Mode 1 uses error detection/correction code (288 bytes).
- Mode 2 uses that space for auxiliary data or as an audio track.

For rewritable CD media, the test can write one or more tracks in one test pass. `cddvdtest` writes tracks on next available space on the media. If the media is full, `cddvdtest` automatically erases the whole media and starts the next test pass from the beginning of the media.

For rewritable DVD media, the test writes only one track in one test pass (because there is only one track in DVD format). `cddvdtest` blanks the media when starting the test, if the media is not already blank.

For rewritable media types, the test verifies write, read, and other supporting functions of CD and DVD RW drives. The supported media include the following:

- CD-R (must be blank)
- CD-RW (can either be blank or contain the SunVTS test data)
- DVD-R (must be blank)
- DVD+R (must be blank)
- DVD-RW
- DVD+RW

Note – `cddvdtest` is not a scalable test.

`cddvdtest` Hardware and Software Requirements

Requirements for `cddvdtest` differ based on the media type.

CD-ROM and DVD-ROM

The drive must have the appropriate CD-ROM or the DVD-ROM media before performing the test.

When a CD-ROM is loaded in the drive, `cddvdtest` uses CD-ROM specific options to test the drive. When a DVD-ROM is loaded, the test uses DVD-ROM specific options. Whenever you change the media in the drive, you must perform a reprobe (refer to the *SunVTS User's Guide* for details) so that the SunVTS kernel associates the correct test options with the media.

CD-RW and DVD-RW

When testing rewritable media, reprobing is required if the media is changed. To prevent accidentally erasing useful data on a media used for testing, `cddvdtest` accepts the test media only if it is blank or it contains SunVTS test data (data that is written by `cddvdtest` itself). The test checks these conditions at probe time and at the start of the test.

In case of nonblank media or media with nonSunVTS test data, the media must be blanked first by using the `cdrw` utility. Because -R or +R media can be written only once, only blank media should be used for write testing. Such media could still run multiple passes of the test because after the first write test, the subsequent tests treat the media as read only and perform the test accordingly.

To prevent media corruption, the test posts a Warning message if it stops druing the write, finalize, format, or erase phases and continues until the current operation is completed. Wait until the test completes before doing any operation on the drive.

Do not stop the test in the middle of a writing operation. Doing so may cause damage to the media in some cases. Set a limited number of passes for `cddvdtest`, set Max Passes=0 (unlimited) and stop the test manually. If a media is damaged, blank the media with `cdrw` command.

Note – DVD+RW media cannot be blanked.

The default delay between two passes for the read write media is three minutes. This setting enables the test to preserve the media by running fewer passes during long test runs. This setting also allows for stopping the test between passes.

cddvdtest Subtests

cddvdtest has different subtests for each media type.

CD-RW and DVD-RW

TABLE 3-1 cddvdtest Subtests for CD-RW and DVD-RW Media

Subtest	Description
Blank	Erases data on DVD-RW media and CD-RW media if not blank. This subtest does not apply to DVD+RW media because this media type can not be blanked.
Simulation	Performs a write track with laser turned off. This mode tests writing function without data written to the media.
Write	Writes to media with predefined data patterns of 0..ff hex. For CD-RW media, track can be specified as data or audio track.
Read	Reads the written data.
Compare	Compares write/read data, reports failure if miscompared.
Read Disk/Track	Reads and shows Disk Table of Contents (TOC).
Eject	Ejects media.

cddvdtest Options

To reach the following dialog box, right-click on the test name in the System Map and select Test Parameter Options. If you do not see this test in the System Map, you might need to expand the collapsed groups, or your system may not include the device appropriate to this test. Refer to the *SunVTS User's Guide* for more details.

cddvdtest has different test options for each media type.

CD-ROM Test Options

This section describes the `cddvdtest` options for CD-ROMs.

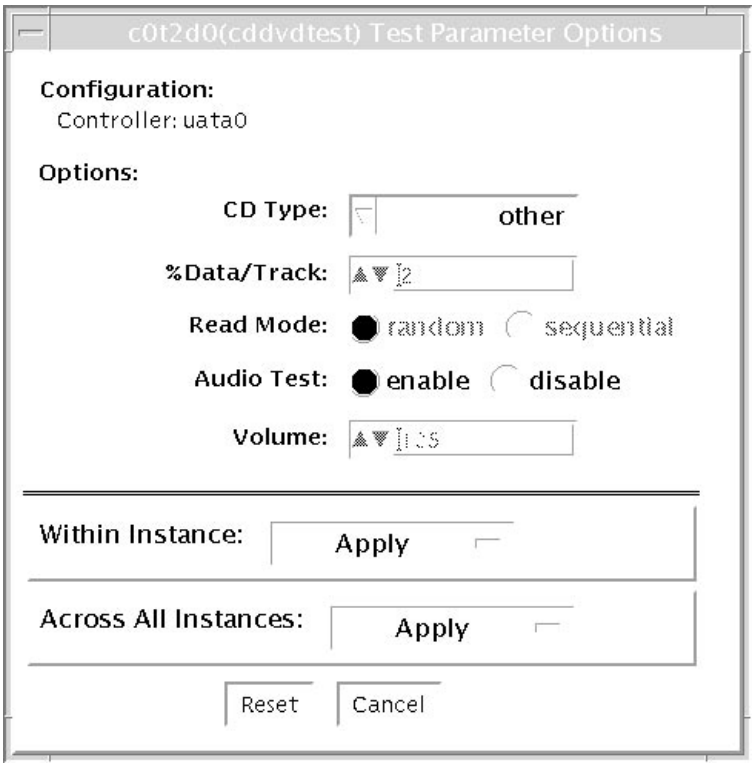


FIGURE 3-1 `cddvdtest` Test Parameter Options Dialog Box for CD-ROM

TABLE 3-2 `cddvdtest` Options for CD-ROM

Option	Description
CD Type	The CD Type menu lists the types of compact discs that can be tested. The choices are: Pdo, Multi-session, or Other the default CD type is other). In Connection test mode, this option has a default value of other. Note - Your choice must correspond with the disc ussed for testing.
% Data/Track	Tests a percentage of data on each track. Type a value between 0 and 100 in this field to indicate the percentage. In the online and connection tests this option has a default value of 2%.
Read Mode	`cddvdtest` reads the CD either in Random or Sequential mode. In Random mode, data blocks are read from random track positions; in Sequential mode, data blocks are read in sequence. For both modes, the total number of blocks read is determined by the %_of_data option. In the online and Connection tests this option has a default value of random.
Audio Test	Enables or disables the audio test. You must connect headphones or a speaker to the audio jack on the CD player to hear audio output. In the Connection test, this option has a default value of disable.
Volume	Adjusts the volume. Type a value between 0 and 255 in this field. In the online and connection tests this option has a default value of 125.

DVD-ROM Test Options

This section describes the cddvdtest options for DVD-ROMs.

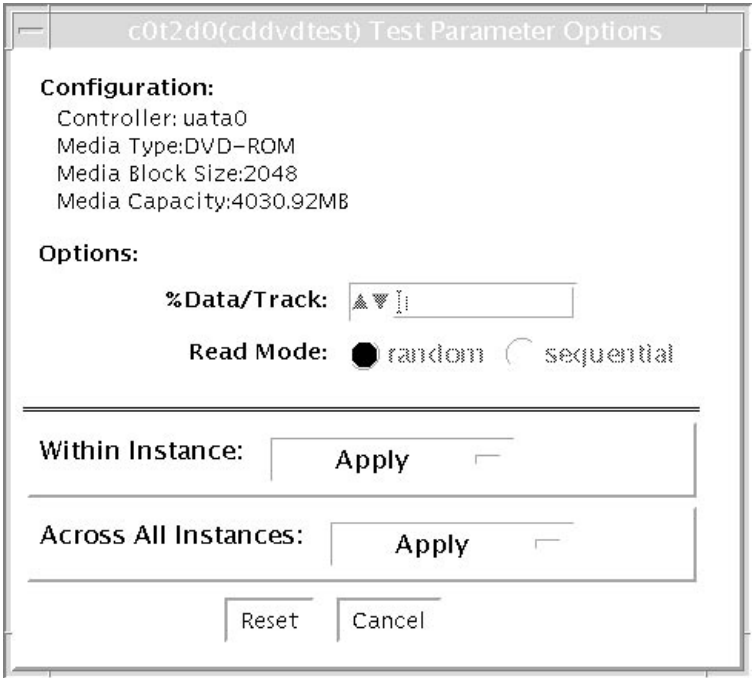


FIGURE 3-2 `cddvdtest` Test Parameter Dialog Box for DVD-ROM

TABLE 3-3 `cddvdrwtest` options for DVD-ROM

Option	Description
% Data/Track	Specifies a percentage of data to test for each track. Use a number between 0 and 100.
Read Mode	Specifies either Random or Sequential mode. Random mode reads data blocks from random track positions. Sequential mode reads data blocks in sequence. For both modes, the total number of blocks read is determined by the % Data/Track value.

CD-RW Test Options

This section describes the test options for CD-RWs.

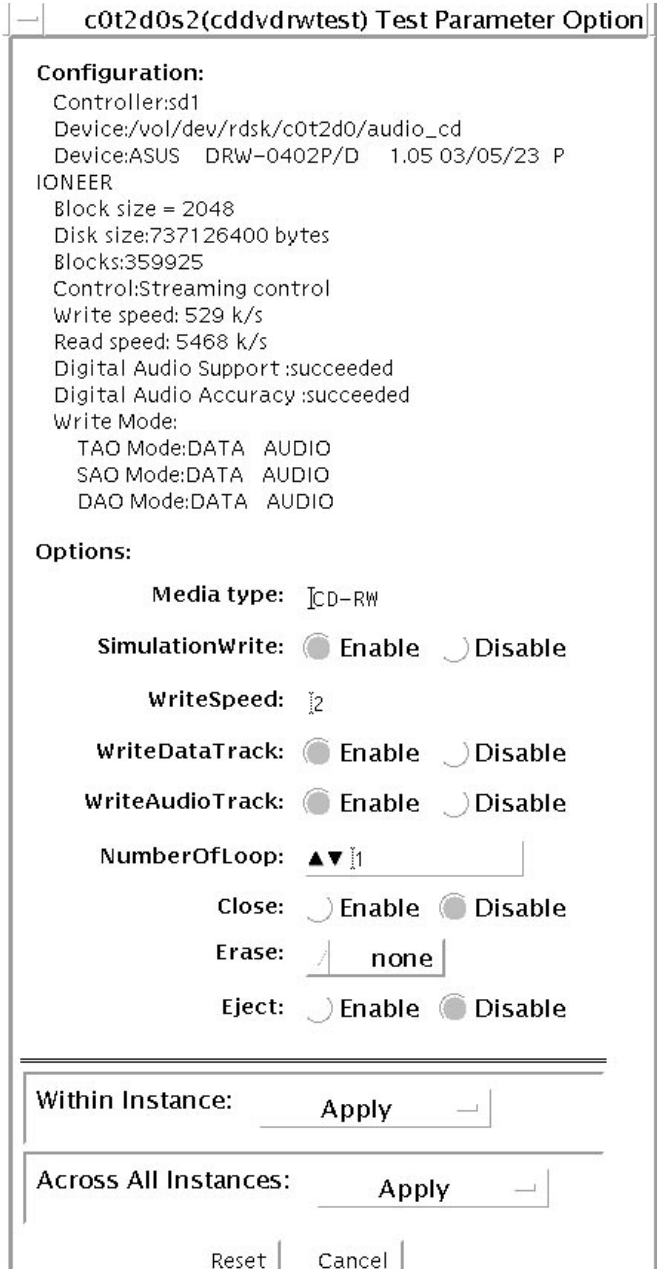


FIGURE 3-3 cddvdtest Test Parameter Options Dialog Box for CD-RW

TABLE 3-4 `cddvdtest` Options for CD-RW

Option	Description
SimulationWrite	Disables or enables simulation write.
WriteSpeed	Speed entered in terms of *nX*. Speed is set to closest approximation of *nX* allowed by the device.
WriteDataTrack	Writes a data track in one test loop.
WriteAudioTrack	Writes an audio track in one test loop.
NumberOfLoop	Number of loops in one test pass comprising data and audio track if enabled.
Close	Closes after writing. Cannot write more tracks on the media.
Erase None	Does not erase. The All option erases the whole disk.
Eject	Ejects the disk when the test is completed. Do not enable this option in a multiple pass test session.

DVD-RW Test Options

This section describes the `cddvdtest` options for DVD-RWs.

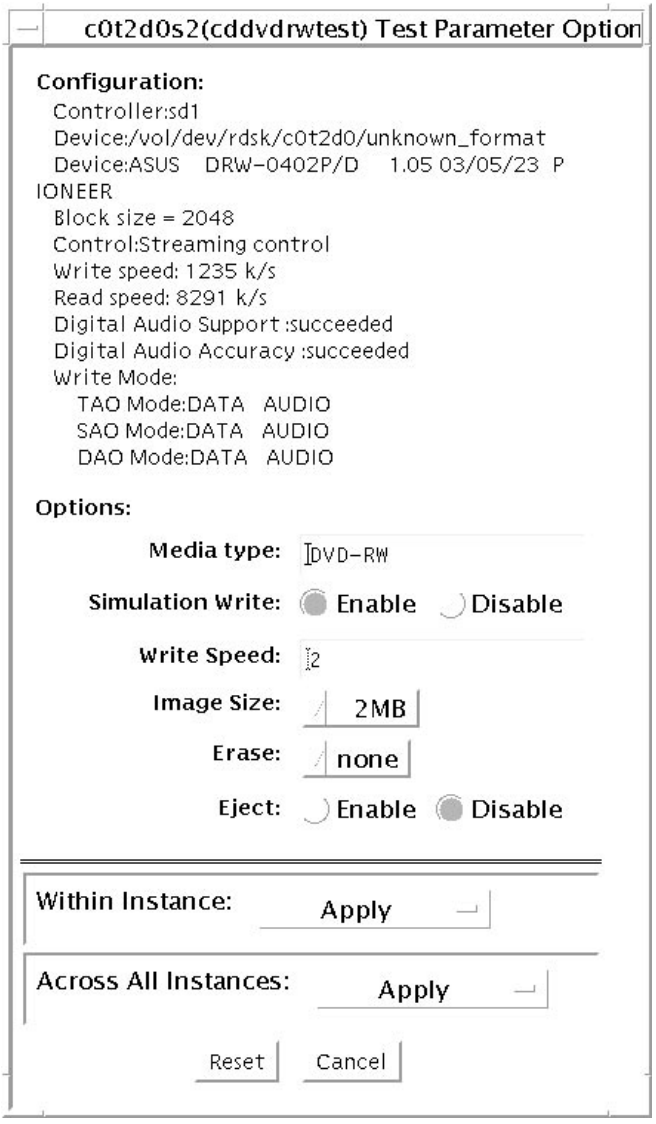


FIGURE 3-4 `cddvdtest` Test Parameter Options Dialog Box for DVD-RW

TABLE 3-5 `cddvdtest` Options for DVD-RW

Options	Description
SimulationWrite	Disables or enables simulation write.
WriteSpeed	Speed entered in terms of *nX*. Speed is set to closest approximation of *nX* allowed by the device.
ImageSize	Specifies track size, how much data is written: 2 MB, 10 MB, 2 GB, 4 GB.
Erase None	Do not erase the media.
Fast	Erases with Fast option. Only the Table of Contents is erased.
All	Erases with All option. Erases whole disk surface.
Eject	Ejects the disk when the test is completed. Do not enable this option in a multiple pass test session.

`cddvdtest` Test Modes

This test supports Connection and Functional test modes for all media types. The test behavior in each mode for the different media types is explained in the following tables.

CD-ROM Test Modes

TABLE 3-6 describes the test modes for CD-ROMs.

TABLE 3-6 Supported Test Modes for CD-ROM

Test Mode	Description
Connection	In this mode, `cddvdtest` verifies that a CD-ROM drive is connected to and configured in the system.
Functional	In this mode, the test registers a failure if the device is found to be busy. This is because SunVTS tests make the assumption that all the resources will be available for testing in the Functional test and the unavailability of a device is interpreted as an indication of a fault condition.

DVD-ROM Test Modes

TABLE 3-7 describes the test modes for DVD-ROMs.

TABLE 3-7 Supported Test Modes for DVD-ROM

Test Mode	Description
Connection	Requests and displays information from the drive and reads two blocks of data from the media to confirm connectivity. An error is reported if no media is loaded in the drive.
Functional	Requests and displays information from the drive, then reads data from the media based on the options that are set in the Test Parameter Options dialog box. An error is reported if no media is loaded in the drive.

CD-RW and DVD-RW Test Modes

TABLE 3-8 describes the test modes for DVD-RWs.

TABLE 3-8 Supported Test Modes for CD-RW and DVD-RW

Test Mode	Description
Connection	Shows basic drive information and supporting modes. Shows disk Table of Contents (TOC).
Functional	The following subtests are done in sequence. Erase - Erases data if DVD-RW media is not blank, or CD-RW media if full. • Simulation (optional) - Performs a write track with Laser turned off. This mode tests writing function without data written to the media. • Write - Writes to media with predefined data patterns of 0..ff hex. For CD media track can be specified as data or audio track. • Read - Reads the written data. • Compare - Compares write/read data, reports failure if miscompared. • Blank media (optional). • Read Disk/Track - Reads and shows disk table of contents. • Ejects (optional).

cddvdtest Command-Line Syntax

cddvdtest has different command-line syntax for each media type.

CD-ROM Command-Line Syntax

/opt/SUNWvts/bin/sparcv9/cddvdtest *standard_arguments* **-o dev=***raw_device_name*, mode=*mode*, read=random|sequential, data=*%_of_data*, vol=*volume*, audio=enable|disable, type=*CD_type*

TABLE 3-9 CD-ROM Command-Line Syntax

Argument	Description
dev=*raw-device-name*	Specifies the name of the raw device to be tested.
read=*random*|*sequential*	Indicates random or sequential read access.
data=*%-of-data*	Sets the percentage of data to be tested. You can specify 0 to 100 percent.
vol=*volume*	Controls the audio volume. You can specify 0 through 255. The default is 255.
audio=*enable*|*disable*	Enables or disables the audio test. You must connect headphones or a speaker to the audio jack on the CD player to hear audio output.
type=*CD-type*	Specifies the type of CD used for the test. The choices are pdo, multi-session, sunos and other. The default is other.

DVD-ROM Command-Line Syntax

/opt/SUNWvts/bin/sparcv9/cddvdtest *standard_arguments* **-o dev=** *device_name*, **read=**random | sequential, **data=**%_*of_data*

TABLE 3-10 DVD-ROM Command-Line Syntax

Argument	Description
dev=*device_name*	Specifies the name of the device to test, for example `/dev/rdsk/cntndn`.
read=random | sequential	Indicates random or sequential read access.
data=%_of_data	Sets the percentage of data to test. Use a number from 0 to 100.

CD-RW Command-Line Syntax

/opt/SUNWvts/bin/sparcv9/cddvdtest *standard_arguments* **-o dev=***cntndnsn*, **media=**CD-RW, nosim, **speed=***n*, nodata, noaudio, **loop=***n*, close, **erase=**{none, all}, eject

TABLE 3-11 CD-RW Command-Line Syntax

Argument	Description
dev=*cntndnsn*	Specifies the device under test.
media=CD-RW	Specifies the media.
nosim	Disables simulation write.
speed=*n*	Specifies the speed. Entere the speed in terms of *nX*.
nodata	Disables data track test.
noaudio	Disables audio track test.
loop=*n*	Specifies 1 to 40, the number of loops in one test pass.
close	Closes track after test after the test, no track can be added.
erase={none, all}	none - Does not erase media after test complete. all - Erases entire disk.
eject	Ejects disk after test completed.

DVD-RW Command-Line Syntax

/opt/SUNWvts/bin/sparcv9/cddvdtest *standard_arguments* **-o dev=***cntndnsn*, **media=**{DVD-RW, DVD+RW}, nosim, **speed=***n*, **imagesize=**{2MB,10MB,2GB,4GB}, **erase=**{none, fast, all}, eject

TABLE 3-12 DVD-RW Command-Line Syntax

Argument	Description
dev=*cntndnsn*	Specifies the device under test.
media={DVD-RW, DVD+RW}	Specifies the media.
nosim	Disables Simulation Write.
speed=*n*	Specifies the speed. Enter the speed in terms of *nX*.
imagesize={2MB,10MB,2GB,4GB}	Specifies the image size used in the write/read test.
erase={none, fast, all}	.none - Does not erase media after test complete. fast - Erases the last track added. all - Erases entire disk.
eject	Ejects the media.

CHAPTER 4

CPU Test (cputest)

cputest Description

The cputest checks specific aspects of SPARC V9 processor datapath functionality.

Note – cputest does support x86 platforms that use the Solaris OS.

The cputest comprises two subtests:

- g0 subtest – tests a processor's g0 register functionality. The g0 subtest is only supported on UltraSPARC-based systems. This subtest is not supported on x86 platforms.
- CUC subtest – Tests a processor's ability to correctly execute a Compress/Uncompress/Compare (cmp) command sequence on machines with the SPARC V9 architecture.
  As the CUC subtest runs, it creates four files in the /tmp/sunvts directory. The CUC subtest uses the following file naming conventions where *nnn* represents the processor unit number and *x* represents a random character string appended to the file name:
  - P*nnn*P*x* – original pattern file (size determined by the cputest File Size option)
  - P*nnn*Z*x* – Compressed version of the pattern file
  - P*nnn*U*x* – Uncompressed data from the P*nnn*Z*x* file

- `P`*nnn*`C`*x* – Comparison data between the `P`*nnn*`P`*x* and `P`*nnn*`U`*x* files

With the `cputest` file retention mode, you can control whether these files are deleted or not, so that in the event of a miscompare, you can view the contents of the files to analyze the miscompared data. The exact names of the files are displayed in the SunVTS message window whenever the files are saved, such as when there is a compression miscompare or when the file retention mode is set to save. Refer to "`cputest` Options" on page 34 for more details.

Caution – Do not run the CUC subtest with the retention mode set to `save` for numerous passes, otherwise the files that are saved in `/tmp/sunvts` can fill the `/tmp` capacity. If `/tmp` is mounted to the swap area, the swap space may become filled to capacity.

Note – Only one instance of `cputest` per processor is possible.

Note – When `cputest` is run with other tests, it may give the error message `exec'd program compress failed with code 1`. This can be an indication of a failed compression program, not necessarily a failed CPU. If this occurs, stop all other tests and run `cputest` alone. If the message occurs again, the CPU is failing.

Note – The `cputest` only runs on SPARC V9 systems.

Note – SunVTS does not support processor sets. If processor sets are defined, you must first delete the processor sets before running SunVTS.

`cputest` Options

To reach the following dialog box, right-click on the test name in the System Map and select Test Parameter Options. If you do not see this test in the System Map, you might need to expand the collapsed groups, or your system might not include the device appropriate to this test. Refer to the *SunVTS User's Guide* for more details.

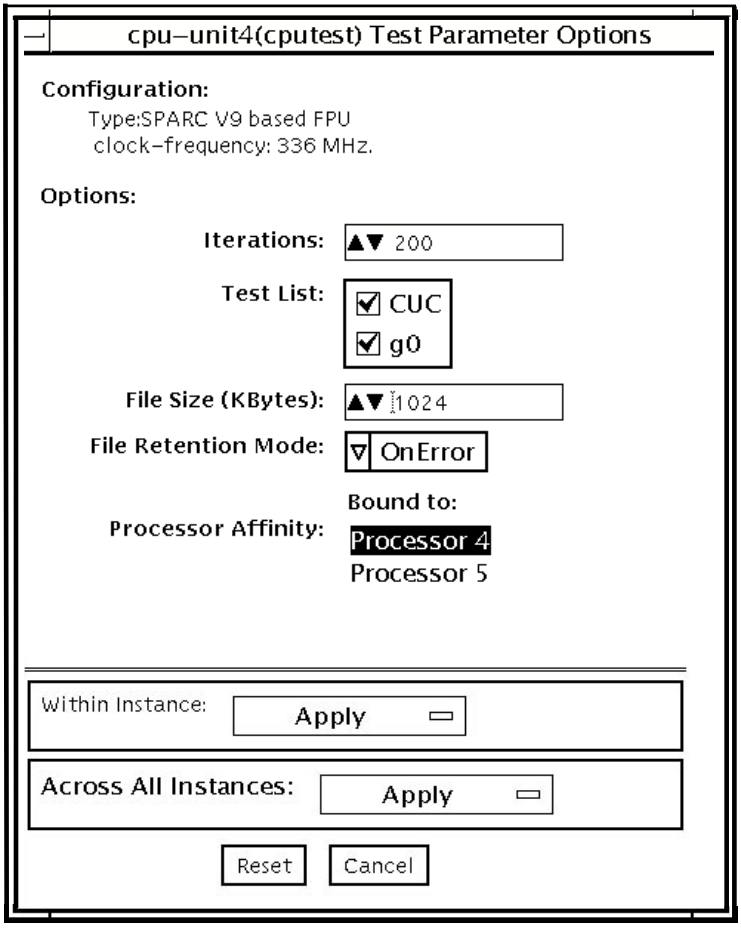


FIGURE 4-1 `cputest` Test Parameter Options Dialog Box

TABLE 4-1 `cputest` Option Dialog Box Descriptions

Option	Description
Iterations	Specifies the number of times to loop on the selected subtests. Use the up and down arrow keys to select a value from 1 to 8192. The default varies depending on the SunVTS test mode.
Test List	Specifies which subtests to run. The choices are: • CUC – The compress/uncompress/compare subtest. • g0 – The g0 register subtest. Note g0 is not supported on x86 platforms. Refer to the general test description at the beginning of this chapter for subtest descriptions. If no subtest is selected, both subtests run.
File Size (KBytes)	Specifies the size of the CUC pattern file in KBytes. Select a value from 1 to 8192. The default varies depending on the SunVTS test mode.
File Retention Mode	Specifies whether the `cputest` removes the CUC pattern files or not. The choices are: • Purge – Unconditionally removes the four subtest files. • Save – Does not remove any of the four subtest files. • OnError – Removes the four subtest files unless the CUC resulted in a miscompare. In this case, do not remove the files. The default is OnError. Refer to the Caution at the beginning of this chapter regarding the Save value.
Processor Affinity	Although the Test Parameter Options dialog box displays the processor affinity Bound To selection box, the processor that corresponds to this instance of the `cputest` is determined when the SunVTS kernel probes for devices. Therefore, switching processor affinity in this dialog box is not supported.

cputest Test Modes

The following table describes how the cputest functions in the different test modes.

TABLE 4-2 cputest Supported Test Modes

Test Mode	Description
Connection	Both subtests are selected. The test options are fixed with the following values: • Iterations=5 • File Size=64 KBytes • File retention=OnError
Functional (Offline)	Both subtests are selectable, and all the test options are available to scale the cputest as needed.
Online	Supported.

cputest Command-Line Syntax

/opt/SUNWvts/bin/cputest *standard-arguments* **-o dev=***device-name***,count=***count-number***,test=***testlist***,size=***file-size*
,retain=*mode*

TABLE 4-3 cputest Command-Line Syntax

Argument	Description
dev=*device-name*	Specifies the name of the device to test, for example, cpu-unit5
count=*count-number*	Defines the number of times to loop on the subtests. Use a number from 1 to 8192. The default is 200.

TABLE 4-3 `cputest` Command-Line Syntax *(Continued)*

Argument	Description
`test=`*testlist*	Specifies which subtests to run. The choices are: • `CUC` • `g0` • `CUC+g0`
`size=`*file-size*	Specifies the size of the CUC pattern file in KBytes. Select a value from `1` to `8192`. The default is `1024`.
`retain=`*mode*	Specifies whether the `cputest` removes the CUC pattern files or not. The choices are: • `Purge` – unconditionally remove the four subtest files • `Save` – do not remove any of the four subtest files • `OnError` – remove the four subtest files unless the CUC resulted in a miscompare. In this case do not remove the files. The default is `OnError`. Refer to the Caution at the beginning of this chapter regarding the `Save` value.

Note – 64-bit tests are located in the `sparcv9` subdirectory: `/opt/SUNWvts/bin/sparcv9/`*testname*, or the relative path to which you installed SunVTS. If a test is not present in this directory, then it might be available as a 32-bit test only. For more information, see "32-Bit and 64-Bit Tests" on page 5.

CHAPTER 5

Cryptographics Test (`cryptotest`)

`cryptotest` Description

The Encryption Framework in Solaris 10 provides a user level API for access to cryptographic accelerators. This API is based on the PKCS#11 standard. Cryptographic accelerators are referred to as PKCS#11 tokens, and each cryptographic algorithm the token accelerates is referred to as a mechanism.

`cryptotest` tests the mechanism supported by PKCS#11 tokens in the Solaris OS.

`cryptotest` supports the Sun Crypto Accelerator 500, Sun Cyrpto Accelerator 1000, Sun Cyrpto Accelerator 4000, and the Niagara Crypto Provider. PKCS documents and information are available at:
`http://www.rsasecurity.com/rsalabs/PKCS`

TABLE 5-1 Definitions of the Mechanisms Tested by `cryptotest`

Algorithm	Description
DSA	Digital signature algorithm
DES	Data encryption standard as defined in FIPS PUB 46-3
MD5 RSA	Data Security MD5 message-digest algorithm

TABLE 5-1 *(Continued)*Definitions of the Mechanisms Tested by `cryptotest`

Algorithm	Description
RSA	Public key cryptosystem
SHA1	Secure hash algorithm
RNG	Random number generator algorithm

`cryptotest` Subtests

TABLE 5-2 cryptotest Subtests

Subtest	Description
DES	Tests DES bulk encryption
3DES	Tests 3DES bulk encryption
RSA	Tests RSA public and private keys
DSA	Tests DSA signature verification
RNG	Tests random number generation

`cryptotest` Options

To reach the following dialog box, right-click on the test name in the System Map and select Test Parameter Options. Because graphics tests can test multiple types of frame buffers, the test name that is displayed will correspond to the particular framebuffer being tested. If you do not see this test in the System Map, you might need to expand the collapsed groups, or your system might not include the device appropriate to this test. Refer to the *SunVTS User's Guide*.

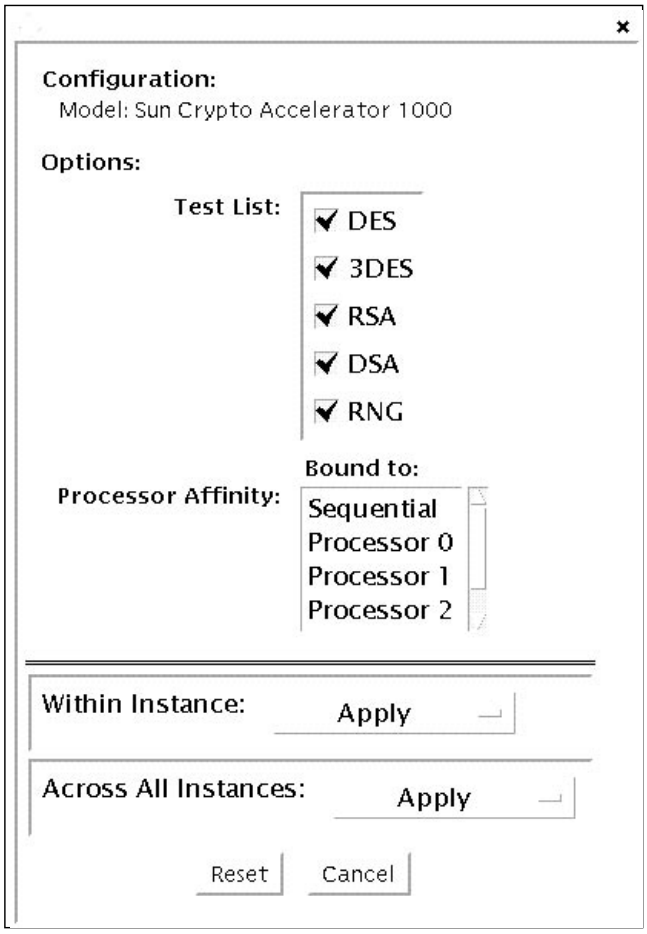


FIGURE 5-1 `dcatest` Test Parameter Options Dialog Box

TABLE 5-3 `dcatest` Options

Option	Description
DES	Tests DES bulk encryption
3DES	Tests 3DES bulk encryption
RSA	Tests RSA public and private keys
DSA	Tests DSA signature verification
RNG	Tests random number generation

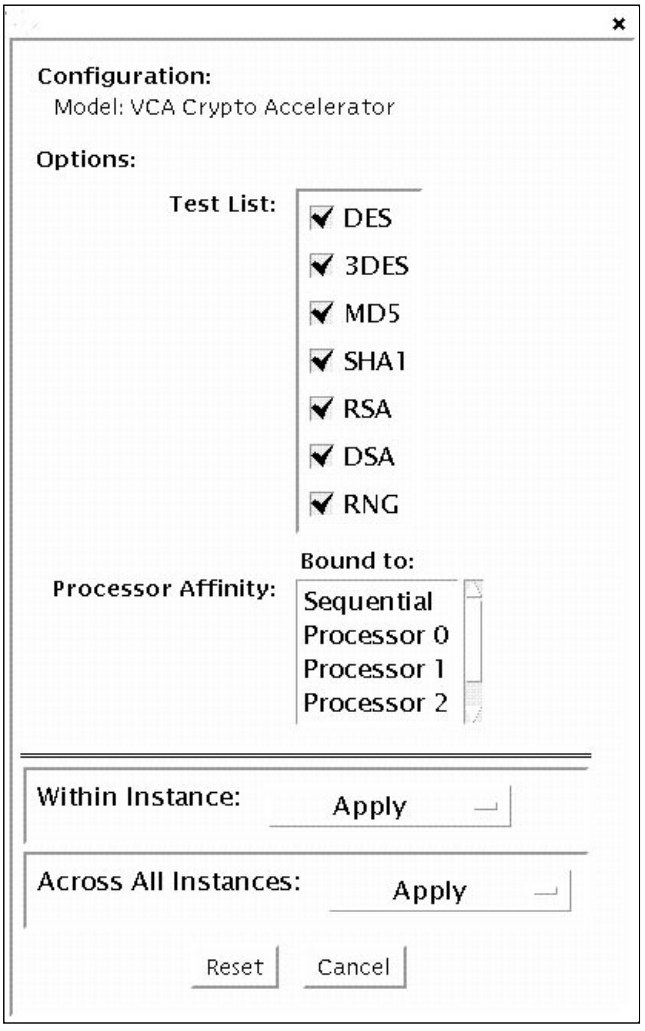


FIGURE 5-2 `vcatest` Test Parameter Options Dialog Box

TABLE 5-4 `vcatest` Options

Option	Description
DES	Tests DES bulk encryption
3DES	Tests 3DES bulk encryption
MD5	Data security MD5 message-digest algorithm.

TABLE 5-4 `vcatest` Options *(Continued)*

Option	Description
SHA1	Secure hash algorithm.
RSA	Tests RSA public and private keys
DSA	Tests DSA signature verification
RNG	Tests random number generation

`cryptotest` Test Modes

TABLE 5-5 `cryptotest` Supported Test Modes

Test Mode	Description
Functional	Runs the full set of tests.

`cryptotest` Command-Line Syntax for `dcatest`

`/opt/SUNWvts/bin/sparcv9/cryptotest -f -o dev=`vca2|dca2**`,tl=`***RSA+DSA***

TABLE 5-6 `cryptotest` Command Line Syntax for `dcatest`

Option	Description
`dev=``dca`*N*	Specifies the instance of the device to test such as `dca0` or `dca2`. Defaults to `dca0` if not included. *N* specifies the placement of the instance number of the device being tested.
`tl=`*testlist*	Specifies the list of subtests to be performed. The subtests for `tl` are separated by the + (plus) character. The supported subtests are DES, 3DES, DSA, RSA, and RNG, so `tl=DES+3DES+DSA+RSA+MD5+SHA1+RNG` enables all subtests. You can also insert `tl=all`, which performs all tests. Defaults to `all` if no subtests are specified.

`cryptotest` Command Line Syntax for `vcatest`

`/opt/SUNWvts/bin/sparcv9/cryptotest -f -o dev=``vca2`**`, tl=`***`RSA+DSA`***

TABLE 5-7 `cryptotest` Command Line Syntax for `vcatest`

Option	Description
`dev=``vca`*`N`*	Specifies the instance of the device to test such as `vca0` or `vca2`. Defaults to `vca0` if not included. *N* specifies the placement of the instance number of the device being tested.
`tl=`*`testlist`*	Specifies the list of subtests to be performed. The subtests for `tl` are separated by the + (plus) character. The supported subtests are DES, 3DES, DSA, RSA, and RNG, so `tl=DES+3DES+DSA+RSA+MD5+SHA1+RNG` enables all subtests. You can also insert `tl=all`, which performs all tests. Defaults to `all` if no subtests are specified.

Note – 64-bit tests are located in the `/bin/64` directory, or the relative path in which you installed SunVTS. If a test is not present in this directory, then it might be available as a 32-bit test only. For more information, see "32-Bit and 64-Bit Tests" on page 5.

CHAPTER 6

Disk and Diskette Drives Test (`disktest`)

`disktest` Description

`disktest` verifies the functionality of hard drives and diskette drives using three subtests (see TABLE 6-1)—Media, File System, and Asynchronous I/O.

Note – `disktest` supports x86 platforms with the Solaris OS.

Most disk drives are supported, including SCSI disks, native or SCSI floppy disks, and IPI drives. The type of drive being tested is displayed at the top of the Test Parameter Options dialog box.

Note – `disktest` is supported on x86 platforms that use the Solaris Operating System. For disks on x86 machines, the disk partitions could range from 0 to 15. `disktest` can be performed on any of these selected partitions.

The `disktest` Test Parameter Options dialog box shows all the partitions that are available for testing. The file System subtest can only be run if the selected partition is mounted. The WriteRead option of the Media subtest is allowed only if a selected partition is *not* mounted.

disktest Test Requirements

By default, disktest does not mount any partitions. To specify that SunVTS premounts all mountable partitions, set the environment variable BYPASS_FS_PROBE to 0 (zero) before starting SunVTS. Premounting can be disabled by disabling BYPASS_FS_PROBE or changing it to a value other than 0 (zero).

The mount point used by disktest is the word *disktest* appended by the name of the disk partition. For example, if the disk partition name is /dev/dsk/c0t3d0s0, disktest mounts the partition as superuser under the name /disktest_c0t3d0s0.

Caution – If a power failure occurs or if disktest is terminated abruptly while the Media subtest is running in WriteRead mode, disk data could possibly be corrupted.

Caution – Running the Media subtest on a disk partition in the WriteRead mode might cause data corruption if the same partition is being used by other applications. Run SunVTS in the offline mode only when there are no other applications running.

disktest tests the disktest drive whether the Volume Management software is running or not. The following mount point names are used:

- If the Volume Management software *is* running, disktest tests the disk drive with the mount point name in the /etc/mnttab file.
- If the Volume Management software *is not* running, disktest tests the disk drive with the device name dev=/dev/diskette. Do not edit the /etc/vold.conf file to change the diskette drives. Currently, the SunVTS software is hard-coded to use these path names as the default logic names.

Loading an option file (refer to the *SunVTS User's Guide* for option file details) that was created when BYPASS_FS_PROBE was set to 0 (zero) might not work if the BYPASS_FS_PROBE environment variable is no longer set to 0. Testing might fail with the following error:

```
SUNWvts.disktest.8088 07/24/98 15:47:22 disktest c0t0d0 FATAL:
"Couldn't get file system information on /disktest_s0t0d0s0,
statvfs() system call failure error: No such file or directory.
```

This error is caused when SunVTS expects to use the predefined mount point names that are created when BYPASS_FS_PROBE is set to 0 (zero), but these mount points do not exist while BYPASS_FS_PROBE is not set to 0.

To use option files with disktest, create two separate option files for the two different states of the BYPASS_FS_PROBE environment variable.

When a large number of disktest instances are run in write/read mode, tests might fail with messages similar to the following.

```
03/22/03 03:33:40 ctech140 SunVTS5.1ps2: VTSID 8011 disktest.FATAL
c1t0d0: "Failed lock mtab semaphore. "semop" system call failure,
errmsg: Invalid argument." Probable_Cause(s): <disktest instances
exceeds system semaphore operation limitation (default system
limit for seminfo_semmnu = 30)><System software error>
Recommended_Action(s): <Add the line "set semsys:seminfo_semmnu=
0x100" to your /etc/system file and reboot the machine> <If the
problem persists, call your authorized Sun service provider.
```

To avoid this issue, add the following entry to the /etc/system file and reboot the system.

```
set semsys:seminfo_semmnu=0x100
```

disktest Subtests

The following table describes the disktest subtests:

TABLE 6-1 disktest Subtests

Subtest	Description
Media subtest	Verifies the disk media by enabling users to run disktest in different modes such as ReadOnly, ReadCompare, and WriteRead. The Media subtest treats the disk partition as one large piece of contiguous data. In WriteRead mode, all instances of disktest communicate through a shared memory service to ensure that they do not overlay the same disk area at the same time. This avoids data corruption. Each of the three modes can run two different methods of disk testings. These are Synchronous I/O and Asynchronous I/O. SyncIO – Reads and writes data using Read/Write system calls in a sequential fashion until the specified percentage of media is covered. AsyncIO – Reads and writes data using aio library calls such as aioread() and aiowrite() until the specified percentage of media is covered. aiowait() is used to synchronize aio operations.
File System subtest	The File system subtest is used to verify the disk file system integrity. It exercises mounted disk partitions carrying the file system. By default, the test only runs on system-mounted partitions, it does not pre-mount any additional partitions. If you want SunVTS to pre-mount all of the unmounted partitions which have a file system, you have to set the environment variable BYPASS_FS_PROBE to '0' (zero). The test creates two temporary files of the size specified by File System File Size, writes the data patterns and compares the two files against each other.

disktest Test Options

To reach the following dialog box, right-click on the test name in the System Map and select Test Parameter Options. If you do not see this test in the System Map, you might need to expand the collapsed groups, or your system may not include the device appropriate to this test. Refer to the *SunVTS User's Guide* for more details.

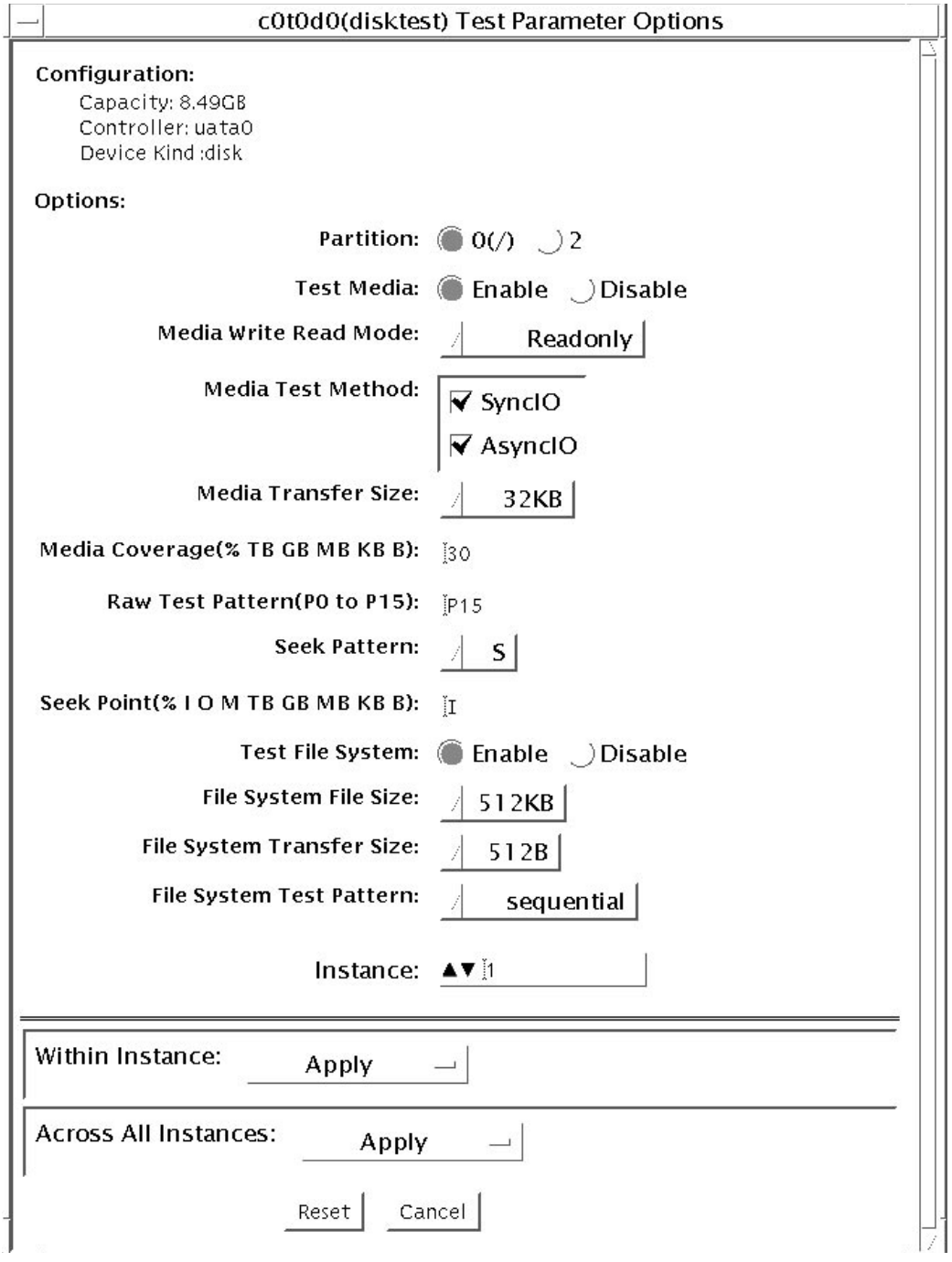


FIGURE 6-1 `disktest` Test Parameter Options Dialog Box

The following table describes the `disktest` option menu for different test modes.

TABLE 6-2 `disktest` Configurations and Options

`disktest` **Options**	**Description**
Partition	Displays the partition for the Media subtest. If a partition is mounted, its mount point is appended after the partition number, such as `1(/usr)`, where `1` is the partition number, and `"(/usr)"` is the mount point.
Test Media	Enables or disables the media subtest.
Media Write Read Mode	Selects Read-Only or Compare after Read, or Read after Write.
Media Test Method	Selects the media test methods (SyncIO and AsyncIO).
Media Coverage (% TB, GB, MB, KB, B)	Tests all or part of a partition (in percentage or in any of TB, GB, MB, KB, B units)
Raw Test Pattern (P0 to P16)	Specifies the write, read pattern. • P0 – Low Frequency Pattern • P1 – Low Transition Density Pattern • P2 – High Transition Density Pattern • P3 – Compliant Jitter Pattern • P4 – Compliant Jitter: RPAT • P5 – Compliant Jitter: CRPAT • P6 – Compliant Jitter: JTPAT • P7 – Compliant Jitter: CJTPAT • P8 – Compliant Jitter: SPAT • P9 – Compliant Jitter: CSPAT • P10 – 8 Bit Cable Pattern • P11 – 16 Bit Cable Pattern • P12 – 8 Bit Xtalk Pattern • P13 – 16 Bit Xtalk Pattern • P14 – MFM Pattern • P15 – Generic Test Patterns • P16 – SATA Test Patterns
Seek Pattern	Specifies the pattern of the disk head movement. • S – Sequential • SR – Sequential Reverse • LS – Low Power Sequential • R – Random • LB – Low Power Butterfly • LR – Low Power Reverse Butterfly • AB – Actuator Butterfly • AR – Actuator Reverse Butterfly

TABLE 6-2 `disktest` Configurations and Options *(Continued)*

`disktest` **Options**	**Description**
Seek Point (%, I, O, M, TB, GB, MB, KB, B)	Specifies the seek point offset for the I/O. You can specify the offset as a percentage or any of TB, GB, MB, KB, B or and I, M, O (Initial, Middle, Outer).
Media Transfer Size	Displays the transfer size of the Media subtest.
Test File System	Selects the File System subtest.
File System File Size	Specifies the size for each of the two temporary files for File System testing.
File System Transfer Size	Displays the transfer size of the file system subtest.
File System Test Pattern	Test pattern of File System subtest.
Connection Test for Hard Disk	• Option Menu for hard disk partition – 0 - 7 [default] • Test Media – [Enable] (fixed to Enable) • Media Write Read Mode – [Read Only] (fixed to Read Only) • Media Test Method-[SyncIO] (fixed to SyncIO) • Media Coverage(%) – 1 • Media Transfer Size – [2 KB] • Test File System – [Disable] (fixed to Disable)

TABLE 6-2 `disktest` Configurations and Options *(Continued)*

`disktest` **Options**	**Description**
Online Mode for Hard Disk	• Partition – 0 - 7 [default] • Test Media – [Enable] [Disable] • Test Mode – [Read-only~] (fixed to Read-only) • Media Coverage (% TB GB MB KB B) – [10~] (fixed to 10%) • Media Transfer Size – [2KB~] (fixed to 2 KB) • Test File System – [Disable~] (fixed to Disable) • Media Test Method – [SyncIO] [AsyncIO] • Raw Test Pattern – [P15~] (fixed to P15) • Seek Pattern – [S~] (fixed to S) • Seek Point (% TB GB MB KB B) – [I~] (fixed to I)
Functional Test for Hard Disk	• Partition – 0 - 7 [default] • Test Media – [Enable] [Disable] • Media Write Read Mode – [Readonly] [CompareRead] [WriteRead] • Media Test method – [SyncIO] [AsyncIO] • Media Coverage (% TB, GB, MB, KB, B) • Raw Test Pattern (P0 to P16) • Media Transfer Size – [2KB] [16KB] [32KB] [64KB] [128KB] [256KB] [512KB] • Test File System – [Enable] [Disable] • File System File Size – [512KB] [2MB] [8MB] [20MB] [100MB] [200MB] • File System Transfer Size – [512B] [1024B] [10KB] [40KB] [80KB] • File System Test Pattern – [sequential] [0x00000000] [0xffffffff] [0x5aa55aa5] [0xdb6db6db] [random] • Seek Pattern – [S~] (fixed to S) • Seek Point (% TB GB MB KB B) – [I~] (fixed to I)
Functional Test for Floppy Disk	• (under Other-Devices group) – Partition: 0 - 7 [default] • Test Media – [Enable] [Disable] • Media Write Read Mode – [Read-only] [BackupWriteRead] • Media Test Method – [SyncIO] [AsyncIO] • Media Coverage (% TB, GB, MB, KB, B) • Raw Test Pattern (P0 to P16) • Media Transfer Size – [2KB] [10KB] [20KB] • Test File System – [Enable] [Disable] • Floppy File Size – [100KB] [200KB] • Floppy Transfer Size – [512B] [1024B] [10KB] • File System Test Pattern – [sequential] [0x00000000] [0xffffffff] [0x5aa55aa5] [0xdb6db6db] [random]

disktest Test Modes

TABLE 6-3 disktest Supported Test Modes

Test Mode	Description
Connection	Only one instance of disktest (which monitors UNIX error messages) is allowed for each disk device. disktest displays messages and reports errors. The test also opens the hard disk, checks the disk configuration, reads a few blocks, and then closes the hard disk. No File System subtest is run. No Write option is available in Connection test mode.
Functional	More than one instance of disktest is allowed for one disk device. The File System subtest, Media subtests, and diskette test can be run in Functional test mode. In Functional mode, disktest performs an additional subtest (Write/Read device buffer subtest) for enclosures.
Online	SunVTS disktest runs the Read Only rawtest with fixed transfer size and fixed rawtest pattern. Both SyncIO and AsyncIO test methods are available. The File system subtest is disabled in the Online test mode. Only one disktest instance could be run in the Online test mode.

disktest Command-Line Syntax

/opt/SUNWvts/bin/disktest *standard-arguments* **-o partition=***0-7 [“(mount-point)”***], rawsub=***E(nable)* | *D(isable)*, **rawrw=***Readonly* | *CompareRead* | *WriteRead*, **rawiosize=***number{...* | *KB* | *kb...}* | *random*, **rawcover=** *number* | *number{TB* | *GB* | *MB* | *KB* | *B* | *tb* | *gb* | *mb* | *kb* | *b}* **rawpattern=***P(0-16)* | *0x8 digit data pattern*, **seekpattern=***{S* | *SR* | *LS* | *R* | *LB* | *LR* | *AB* | *AR}*, **seekpoint=**

{i | m | o | number}, **method=***AsyncIO+SyncIO*, **fssub=***E(nable) | D(isable)*, **fssize=***number{K | KB | M | MB | k | kb | m | mb}*, **fsiosize=***number{K | KB | B | k | kb | b}*, **fspattern=***data-pattern*, **dev=***device-name*

TABLE 6-4 `disktest` Command-Line Syntax

Argument	Description
partition=*0-7 [“(mount-point)”]*	Specifies the partition number as follows: • *n* – The partition number (slice number), usually *0-7* for SPARC and *0-16* for x86 • *mount-point* – The mount point for the mounted partition that you plan to test For example: **partition=6”(/export)”**
rawsub= *E(nable)	D(isable)*	Enables or disables the Media subtest. For example: **rawsub=** *Enable*
rawrw= *Readonly	CompareRead	WriteRead*	Specifies the Media subtest Read, Compare, and Write mode: • Read only • Read twice, Compare (works only with SyncIO method) • Write, Read, Compare, Restore For example: **rawrw=***ReadOnly*
rawiosize= *number{...	KB	kb...}	random*	Specifies the media size to transfer. The block size can be specified in kilobytes. For example, 2K to 512K. For example: **rawiosize=**9
rawcover= *number	number{TB	GB	MB	KB	B	tb	gb	mb	kb	b}*	Specifies media coverage from 0-100 (percentage) of the partition. Media Coverage can also be specified in the following units—TB, GB, MB, KB and B. For example: **rawcover=***40* or **rawcover=***4GB*

TABLE 6-4 `disktest` Command-Line Syntax *(Continued)*

Argument	Description
`rawpattern=`*P(0-16)* | *0x8 digit data pattern*	`rawpattern` could be specified as a predefined pattern set, *P(0-16)*, or an 8 digit pattern could be specified as: *0xaa55aa55+0xff00ff00+0x*. The following is a description of the supported pre-defined patterns: • P0 – Low Frequency Pattern • P1 – Low Transition Density Pattern • P2 – High Transition Density Pattern • P3 – Compliant Jitter Pattern • P4 – Compliant Jitter: RPAT • P5 – Compliant Jitter: CRPAT • P6 – Compliant Jitter: JTPAT • P7 – Compliant Jitter: CJTPAT • P8 – Compliant Jitter: SPAT • P9 – Compliant Jitter: CSPAT • P10 – 8 Bit Cable Pattern • P11 – 16 Bit Cable Pattern • P12 – 8 Bit Xtalk Pattern • P13 – 16 Bit Xtalk Pattern • P14 – MFM Pattern • P15 – Generic Test Patterns • P16 - SATA Test Patterns For example: **`rawpattern=`***P1*
`seekpattern=` *{S | SR | LS | R | LB | LR | AB | AR}*	Selects the type of seek test to run on the disk drive. `disktest` supports the following pattern types: • `S` – Sequential • `SR` – Sequential Reverse • `LS` – Low Power Sequential • `R` – Random • `LB` – Low Power Butterfly • `LR` – Low Power Reverse Butterfly • `AB` – Actuator Butterfly • `AR` – Actuator Reverse Butterfly For example: `seekpattern=`*S*

TABLE 6-4 `disktest` Command-Line Syntax *(Continued)*

Argument	Description
seekpoint=*{i | m | o | number}*	Specifies the seek-point for the I/O. This could be specified either in terms of the range - inner, middle and outer. Or in terms of absolute seek location. The absolute location is specied by a number followed by any of the following units {TB | GB | MB | KB | B | tb | gb | mb | kb | b}. For example: • `seekpoint=`*I*, starts the I/O from block 1. • `seekpoint=`*M*, starts the I/O from middle offset of the partition.
method=*AsyncIO+SyncIO*	Specifies the Media access method. You can choose to use either or both methods . If you use both access methods together, you must insert a '+' between the two: • AsyncIO: Runs the asynchronous I/O test, using the async read/write feature of the Solaris disk driver • SyncIO: Runs the synchronous I/O test. For example: **method=***AsyncIO*
fssub=*E(nable) | D(isable)*	Enables or disables the File System subtest. File System subtest runs on a mounted partition with a File System.
fspattern=*data-pattern*	Specifies the file system data pattern as sequential or random or one of the patterns selected from the list. *{seq(uential) | 0x0(0000000) | 0xf(ffffffff) | 0xa (5a5a5a5) | 0x5(a5a5a5a) | ran(dom) | 0xd(b6db6db)}* For example: • **fspattern=***0xa* • **fspattern=***seq*
fssize= *number{K | KB | M | MB | k | kb | m | mb}*	Indicates the file system subtest size in Megabytes or Kilobytes: • `K | k | KB | kb` – kilobytes • `M | m | MB | mb` – megabytes `512KB | 2MB | 8MB | 20MB | 100MB | 200MB`
fsiosize= *number{K | KB | B | k | kb | b}*	Indicates the size of the file system subtest I/O transfer in bytes or Kilobytes: • B | b – bytes • `K | k | KB | kb` – Kilobytes `512B | 1024B | 10KB | 40KB | 80KB`
dev=*device-name*	Specifies the name of the disk to be tested. For example: c0t3d0.

The following example shows how to run `disktest` on a partition "0" (which is mounted under "/") for the disk device `c0t0d0`. The media subtest is enabled in ReadOnly mode using the SyncIO method. The coverage specified is 30% with 512 KB transfer size. The File System subtest is disabled.

```
# /opt/SUNWvts/bin/disktest -f -o partition=0"(/)", rawsub=Enable,
rawrw=ReadOnly, method=SyncIO, rawcover=30, rawiosize=512KB,
fssub=Disable, dev=c0t0d0
```

Note – 64-bit tests are located in the `/bin/64` directory, or the relative path in which you installed SunVTS. If a test is not present in this directory, then it might be available as a 32-bit test only. For more information, see "32-Bit and 64-Bit Tests" on page 5.

CHAPTER 7

Data Translation Look-Aside Buffer (`dtlbtest`)

`dtlbtest` Description

The `dtlbtest` verifies the proper functioning of DTLBs, but it does not measure performance of DTLBs.

Note – `dtlbtest` supports x86 platforms on Solaris OS.

This test verifies the following function of DTLBs:

- Hit/miss tests – Stress DTLBs by generating large numbers of TLB hits and misses.
- Address Pattern tests – Verify that DTLBs correctly translate virtual addresses to physical addresses with stressful address patterns.

`dtlbtest` detects the installed CPU type and iterprets the CPU architecture.

`dtlbtest` is classified as an exclusive SunVTS test because its accuracy and coverage depend significantly on the system's background *quietness*. For the best result, avoid performing any other SunVTS tests in the background, and also disable all user processes and nonessential daemons.

`dtlbtest` Options

To reach the following dialog box, select the exclusive test mode and right-click on the test name in the System Map and select Test Parameter Options. If you do not see this test in the System Map, you might need to expand the collapsed groups, or your system may not include the device appropriate to this test. Refer to the *SunVTS User's Guide* for more details.

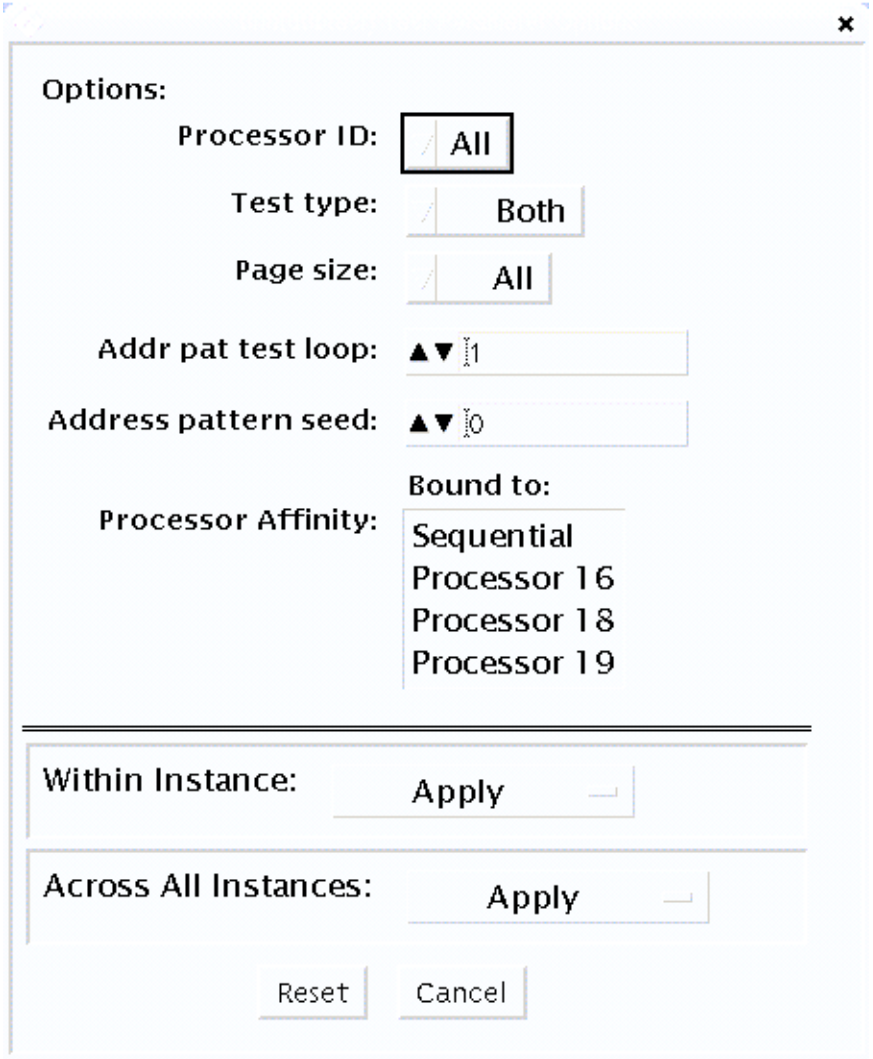


FIGURE 7-1 `dtlbtest` Test Parameter Options Dialog Box

TABLE 7-1 dtlbtest Options

Option	Description
Processor ID	Specifies the CPU ID for testing. The default value is all.
Test Type	Specifies the type of test need to be performed, Hit/miss, address pattern, or both. The default value is both.
Page Size	Specifies the DTLB page size to test. The default value is all. Possible page size values are 8k, 64k, 512K, and 4M.
Address Pattern loop count	Specifies the address pattern loop count. The default value is 1. Loop count value ranges between 1 and 1000.
Address pattern seed value	Specifies the address pattern seed value. The default value is 0.

TABLE 7-2 `dtlbtest` Supported Test Modes

Test Mode	Description
Exclusive	Performs only the `dtlbtest` (full test).

`dtlbtest` Command-Line Syntax

`/opt/SUNWvts/bin/sparcv9/dtlbtest [-scruvdtlxnf] [-p n] [-i n] [-w n] [-o cpu=`*N*`, test=`***x*`, psize=`***x*`, aloop=`***n*`, aseed=`***n*`]**

TABLE 7-3 `dtlbtest` Command-Line Syntax

Argument	Description
`cpu=`*cpuid*	Specifies the CPU ID. The default value is all.
`test=`*testType*	Specifies the type of test h, a, or both. The default value is both.
`psize=`*pageSize*	Specifies the page size to be tested. The default value is all.
`aloop=`*loopCount*	Specifies the address pattern loop count. The default value is 1.
`aseed=`*seedValue*	Specifies the address pattern seed value. The default value is 0.

CHAPTER 8

Emulex HBA Test (`emlxtest`)

`emlxtest` verifies the proper functioning of the Emulex LP10000 Dual and Single Port 2 Gigabyte Fibre Channel PCI host bus adapters (HBAs). The supported HBAs are 133 MHz PCI-X 2 Gigabyte Fibre Channel based on the Emulex LP10000-S (Amber-2r) and LP10000DC-S (Crystal-2r) HBAs, which are the single and dual channel versions of the Emulex LP10000 line of products.

Note – `emlxtest` is supported on both Sun Ultra SPARC and x86 AMD64 architectures.

Note – `emlxtest` must be performed as a standalone test. Ensure that no other tests are running at the same time as the `emlxtest`.

`emlxtest` Options

To reach the following dialog box, right-click on the test name in the System Map and select Test Parameter Options. If you do not see this test in the System Map, you might need to expand the collapsed groups, or your system might not include the device appropriate to this test. Refer to the *SunVTS User's Guide* for more details.

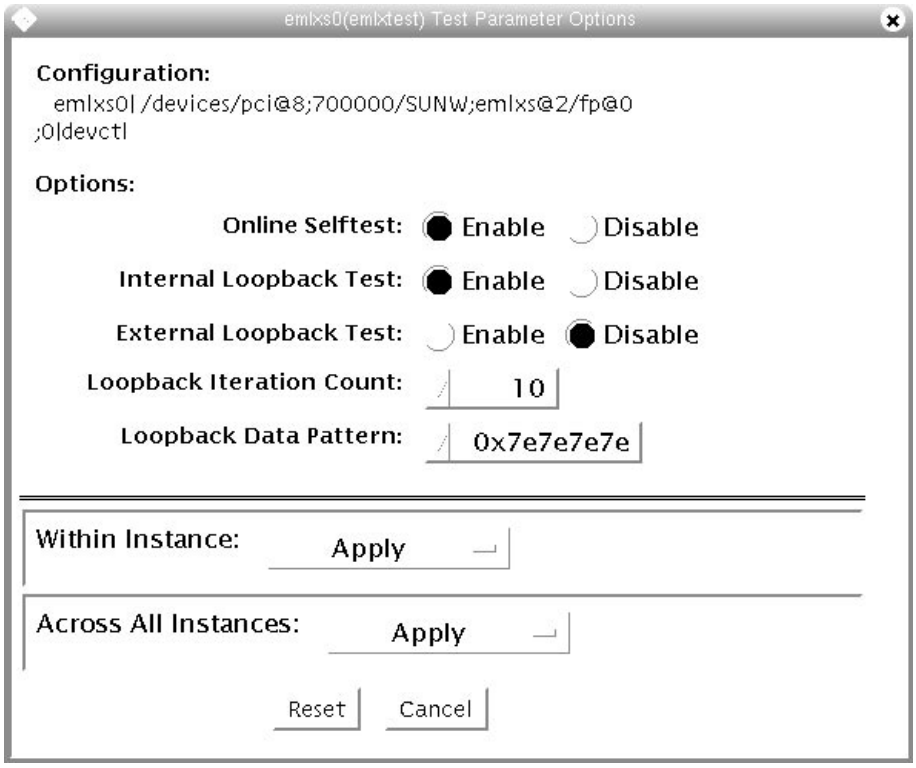


FIGURE 8-1 `emlxtest` Test Parameter Options Dialog Box

The following table lists the test parameter options for `emlxtest`.

TABLE 8-1 `emlxtest` Options

`emlxtest` **Options**	**Description**
Online Selftest	Evaluates the functionality of ISP hardware by performing the following tests: • Transmit FIFO test • Receive FIFO test • SRAM test • Misc. Register tests Run by default, but can be deselected.
Internal Loopback Test	Performs internal loopback test within the host adapter ISP hardware at the interface. This test is done with data sourcing from the system memory and going to the system memory. The desired data pattern, transfer length, and iteration count can be selected with the test parameters menu. Run by default, but can be deselected.
External Loopback Test	Performs an external loopback test. This test is performed with data sourcing from the system memory and going to the system memory. The desired data pattern, transfer length, and iteration count can be selected via the test parameters menu. This is an intervention test, because a loopback cable is needed from the transceiver to the receiver of the Emulex port when testing this port by itself. This subtest can also test the entire fibre channel loop when the loop is connected to the storage to be tested. Not run by default, but can be selected.
Loopback Iteration Count	Sets the number of times to loop the internal 10-bit, internal 1-bit, and external loopback tests. Default value is `10`.
Loopback Data Pattern	Selects the data pattern to loop for the internal 10-bit, internal 1-bit, and external loopback tests. Default value is `0x7e7e7e7e`.

emlxtest Test Modes

TABLE 8-2 emlxtest Supported Test Modes

Test Mode	Description
Connection	Opens and closes the Emulex port.
Exclusive	Performs the full set of tests.

emlxtest Command-Line Syntax

/opt/SUNWvts/bin/emlxtest *standard-arguments*
-x -v -o dev=*device-name***,run-connect=***Yes* | *No***,selftest=***Enable* | *Disable* , **ilb=***Enable* | *Disable* , **elb=***Enable* | *Disable* , **lbfcount=***iteration-count* , **lbfpattern=***hex-pattern*

Note – The -x option must be used from the command line.

TABLE 8-3 emlxtest Command-Line Syntax

Argument	Description
dev=*device-name*	The name of the device to test.
selftest=*Enable*	*Disable*	Enables or disables the selftest command. Evaluates the functionality of the ISP hardware. Enabled by default.
ilb=*Enable*	*Disable*	Enables or disables the internal test. Performs internal loopback test within the host adapter ISP hardware at the 1-bit interface. Enabled by default.
elb=*Enable*	*Disable*	Enables or disables the external loopback test. The desired data pattern, transfer length, and iteration count can be selected on the test parameters menu. Requires a cable for this intervention test. Disabled by default.
lbfcount=*iteration-count*	Controls the number of times the loopback test will run, for example, 100. Default value is 10.
lbfpattern=*hex-pattern*	Lists the data pattern to loop, for example, 0x7E7E7E7E. Default value is 0x7E7E7E7E.

Note – 64-bit tests are located in the `sparcv9` subdirectory: `/opt/SUNWvts/bin/sparcv9/`*testname, or the relative path to which you installed SunVTS*. If a test is not present in this directory, then it may only be available as a 32-bit test. For more information refer to the "32-Bit and 64-Bit Tests" section of the *SunVTS 5.1 Test Reference Manual* (816-5145-10).

CHAPTER 9

Floating Point Unit Test (`fputest`)

`fputest` Description

The `fputest` checks the functionality of the floating point unit in a Sun SPARC based CPU. The test verifies the functionality by various arithmetic operations. In addition, the `fputest` stresses the CPU with the use of benchmarks. Both single and double precision numbers are used for the operations.

Note – `fputest` supports x86 platforms on Solaris.

Note – Three benchmarks of `fputest`—`cparanoia`, `kcdiv`, and `kcsqrt`—have been ported over for x86 systems. The x86 version of `fputest` only uses the Double Precision operations for rounding, chopping, sticky bit, and so on.

When `fputest` is chosen in Exclusive test mode from the SunVTS graphical user interface, it may run multiple instances in parallel on different CPUs. The number of such instances that may be running in parallel at the same time is dynamically determined depending on system resources.

`fputest` Subtests

Instruction tests include:

- FSR Register test
- Registers test
- NACK test
- Move Registers test
- Positive to Negative test
- Negative to Positive test
- Absolute test
- Single-Precision Integer to Floating Point test
- Double-Precision Integer to Floating Point test
- Single-Precision Floating Point to Integer test
- Double-Precision Floating Point to Integer test
- Single-Precision Round Toward Zero test
- Double-Precision Round Toward Zero test
- Single to Double-Precision Format Conversion test
- Double to Single-Precision Format Conversion test
- Single and Double-Precision Addition, Subtraction, Multiplication, Square-root, Division, and Compare tests
- Single and Double-Precision Compare and Exception if Unordered tests
- Branching and No Branching on Condition Instructions tests
- Single and Double-Precision Chaining tests
- Weitek Status tests
- Lock test
- Single and Double-Precision Datapath tests
- Timing (load) test

Benchmark tests include:

- Lapack test
- Cparanoia test
- Kcsqrt test
- Kcdiv test
- Clorenz test
- Cvector test

`fputest` Options

To reach the following dialog box, right-click on the test name in the System Map and select Test Parameter Options. If you do not see this test in the System Map, you might need to expand the collapsed groups, or your system might not include the device appropriate to this test. Refer to the *SunVTS User's Guide* for more details.

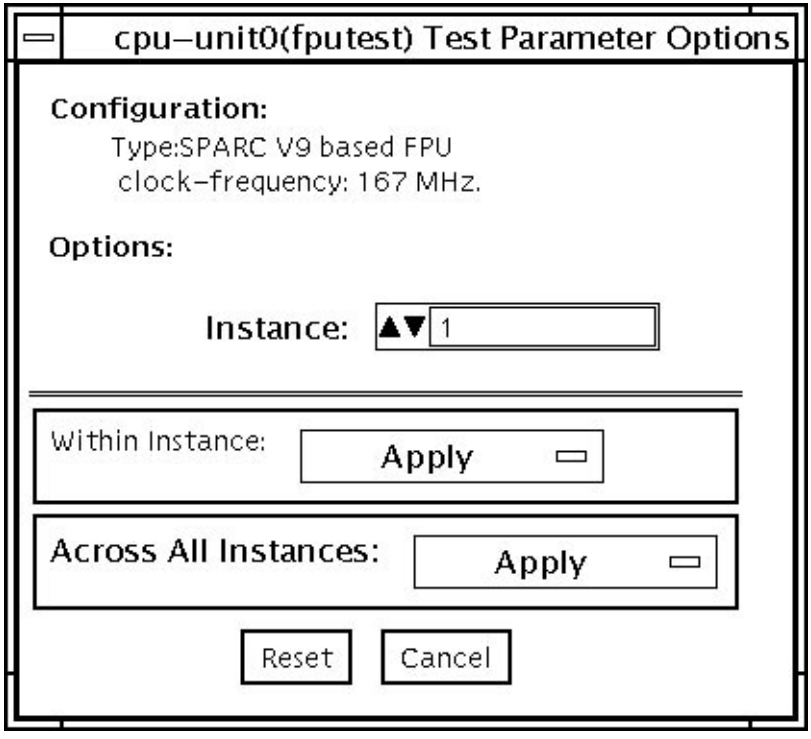


FIGURE 9-1 `fputest` Test Parameter Options Dialog Box

Note – It is not advisable to use the Processor Affinity option for this test. Doing so reduces the effectiveness of the test.

---

fputest Test Modes

TABLE 9-1 fputest Supported Test Modes

Test Mode	Description
Connection	Includes all the instruction tests.
Functional	Performs all the instruction tests and all the benchmark tests.
Exclusive	Performs several FPU benchmark tests.
Online	Supported.

fputest Command-Line Syntax

/opt/SUNWvts/bin/fputest [**-scruvdtlxnf**] [**-p n**] [**-i n**] [**-w n**] [**-o dev=cpu-unit***N*]

Note – Options -s, -p, -i, and -w are not applicable when tests are invoked from the command line.

TABLE 9-2 fputest Command-Line Syntax

Argument	Description
-s	Enables SunVTS mode.
-c	Enables core file.
-r	Enables run on error.
-u	Enables list usage.
-v	Enables Verbose mode.
-d	Enables Debug mode.
-t	Enables test function trace mode.
-l	Enables Online mode, and forks tests automatically against all online CPU IDs in the system.
-x	Enables Exlusive mode, and forks tests automatically against all online CPU IDs in the system.

TABLE 9-2 `fputest` Command-Line Syntax

Argument	Description
`-n`	Enables Connection mode.
`-f`	Enables Offline mode, and forks tests automatically against all online CPU IDs in the system.
-p n	*n* specifies the number of passes. The default is 1.
`-i` *n*	*n* specifies the number of total instances for the test. The default is 1.
`-w` *n*	*n* specifies which instance this test is assignes. The default is 0.
`-o`	Enables test specific command arguments.
`dev=cpu-unit`*N*	Specifies the CPU unit to be tested. *N* specifies the numeric ID of online CPU. This is not a required option.

Usage Examples:

Note – When using `fputest` specific arguments, the `-o` and `dev` options are required.

To execute exclusive `fputest` with verbose messages against CPU 72, use the following syntax:

```
# /opt/SUNWvts/bin/fputest -xvo dev=cpu-unit72
```

To execute functional `fputest` with verbose messages against CPU 3, use the following syntax:

```
# /opt/SUNWvts/bin/fputest -fvo dev=cpu-unit3
```

Note – 64-bit tests are located in the `sparcv9` subdirectory: `/opt/SUNWvts/bin/sparcv9/`*testname*, or the relative path to which you installed SunVTS. If a test is not present in this directory, then it might be available as a 32-bit test only. For more information, see "32-Bit and 64-Bit Tests" on page 5.

CHAPTER 10

Infiniband Host Channel Adapter Test (ibhcatest)

ibhcatest Description

ibhcatest comprises multiple iRISC CPU cores, two 4x Infiniband ports, and integrated SerDes components. In addition, the ibhcatest external associated components include FLASH ROM and DDR memory. ibhcatest provides high speed interconnect through PCI interface to external Infiniband fabric. Supported platforms include: two 1U and two 2U x86 AMD Opteron entry-level servers, Sun Fire V2XX, V4XX and E-series high-end servers.

Note – ibhcatest supports the same set of options for both SPARC and x86 platforms.

ibhcatest exercises and verifies the proper operation of the Tavor chip and its associated components, such as DDR memory, flash PROM, and internal IB packet transmit/receive circuitry. The isolates single faults to the identifiable component(s).

ibhcatest supports three execution test modes in SunVTS Connection, Exclusive and Functional. In Connection mode, the test queries for the Tavor firmware and hardware revision, and running internal loopback.

The internal loopback test is run at least once depending on the amount of time each pass takes. In Functional mode all subtests are executed according to the options selected. In Exclusive mode all subtests are executed in sequence.

Tavor supports an internal loopback mechanism that is very similar to the actual operation. The main difference is that data does not go through the integrated SerDes and the 4x IB port circuitry. On the receiving side, data does not get verified by the CRC algorithm. Otherwise, all other components of Tavor that involved in transmitting and receiving data packets are being exercised by `ibhcatest`.

Tavor-based HCA uses a single, 256 MB DDR memory for data storage at run time. This data storage is shared by three interdependent clients, Tavor driver, firmware, and hardware. During driver initialization, predetermined data structures and data are laid out in the memory.

With no exclusive access from the driver side, subsequent writes to any memory location that contain real data can cause undesirable results like a system crash. Furthermore, the data allocation size is fixed, writing to the remaining free memory does not add any value in terms of finding faults.

The memory subtest is limited to read only operations which cover the entire DDR memory. The test does not check for data corruption and no mechanism for triggering bit errors through writing to memory. The test uncovers bus related problems.

ibhcatest Subtests

TABLE 10-1 `ibhcatest` Subtests

Subtest	Description
Internal Loopback Test	The HCA supports internal loopback for packets transmitted between QPs that are assigned to the same HCA port. If a packet is being transmitted to a DLID that is equivalent to the Port LID with the LMC bits masked out or the packet DLID is a multicast LID, the packet goes on the loopback path. In this latter case, the packet also is transmitted to the fabric. When a packet is looped back, it must pass the SL2VL mapping. If the mapping yields 15 or a nonoperational VL, the packet is discarded. In the inbound direction, the ICRC and VCRC checks are blindly passed for looped back packets. Note that internal loopback is supported only for packets that are transmitted and received on the same port. Packets that are transmitted on one port and received on another port are transmitted to the fabric. The fabric directs these packets to the destination port. This subtest uses interfaces from the Tavor driver to perform loopback testing. Information such as data pattern for data packets, port number, CQ polling, retries between iteration, and the number of iterations for each `ioctl` call are passed to the driver. Once finished, status regarding the number passes completed is returned. If the number of passes does not match the number of iterations, a failure has occurred. This might happen when the number of retries is exhausted, and the last failing buffer in the retry series is returned as result. SunVTS then determines exactly what failed in the buffer and reports the failure. The options for this subtest are as follows: • `lb=`*Enabled*`|`*Disabled* – Turn the loopback test on or off. • `tlbport=`*1+2* – Loopback test on Port 1 and/or 2, default is 1+2. • `data=`*Pattern* – Specific data pattern (default patterns 0xa5a5a5a5). • `cq=`*Time* – Number of CQ polling time value (in microseconds per iteration). The default is 55000. The maximum is 1000000. • `loop=`*Number* – Number of loopback iterations for each pass. The default is 200. The maximum is 1000. • `warn=`*Enabled*`|`*Disabled* – When enabled, prints a warning message.
DDR READ Test	This subtest comprises two test modes, Sequential and Random. The start and end address offsets are determined dynamically by obtaining them from the firmware. In Sequential mode, length and starting offset are instructed by the test option `rdoffset` and `rdsz`. Then the test sequentially reads data from each memory address until all memory locations are covered or the end address is reached. Each read is accomplished by an `ioctl` call to the driver. The test returns pass or fail based on the completion status of each `ioctl` call. In Random mode, this subtest reads the number of `rdsz` times in a randomly generated address bound by the start and end address offset. The options for this subtest are as follows: • `ddr=`*Enabled*`|`*Disabled* – Turns the DDR memory test on or off. • `rdoffset=`*Offset* – Starts offset of DDR memory read. The default is 0x0 in hexadecimal • `rdsz=`*Size* – Read number of byte of DDR memory from Offset to Max address location. The default is 0x2000. The maximum is 256 MB in hexadecimal

Caution – In the SunVTS environment, `ibhcatest` and `nettest` are mutually exclusive. `nettest` has higher priority if the IB port interface is plumbed up when SunVTS is invoked. These two tests cannot be run at the same time at the command line If both of these tests are invoked at the command line, `ibhcatest` exits gracefully if the IB port interface is plumbed up. The commands to bring down the IB daemon (`ibd` [`IPoIB`]) are as follows:

```
# ifconfig ibdXX down
# ifconfig ibdXX unplumb
```

Where *XX* is the instance number of the interface.

`ibhcatest` Options

To reach the following dialog box, right-click on the test name in the System Map and select Test Parameter Options. Because graphics tests can test multiple types of frame buffers, the test name that is displayed corresponds to the particular frame buffer being tested. See "Testing Frame Buffers" on page 10. If you do not see this test in the System Map, you might need to expand the collapsed groups, or your system might not include the device appropriate to this test. Refer to the *SunVTS User's Guide*.

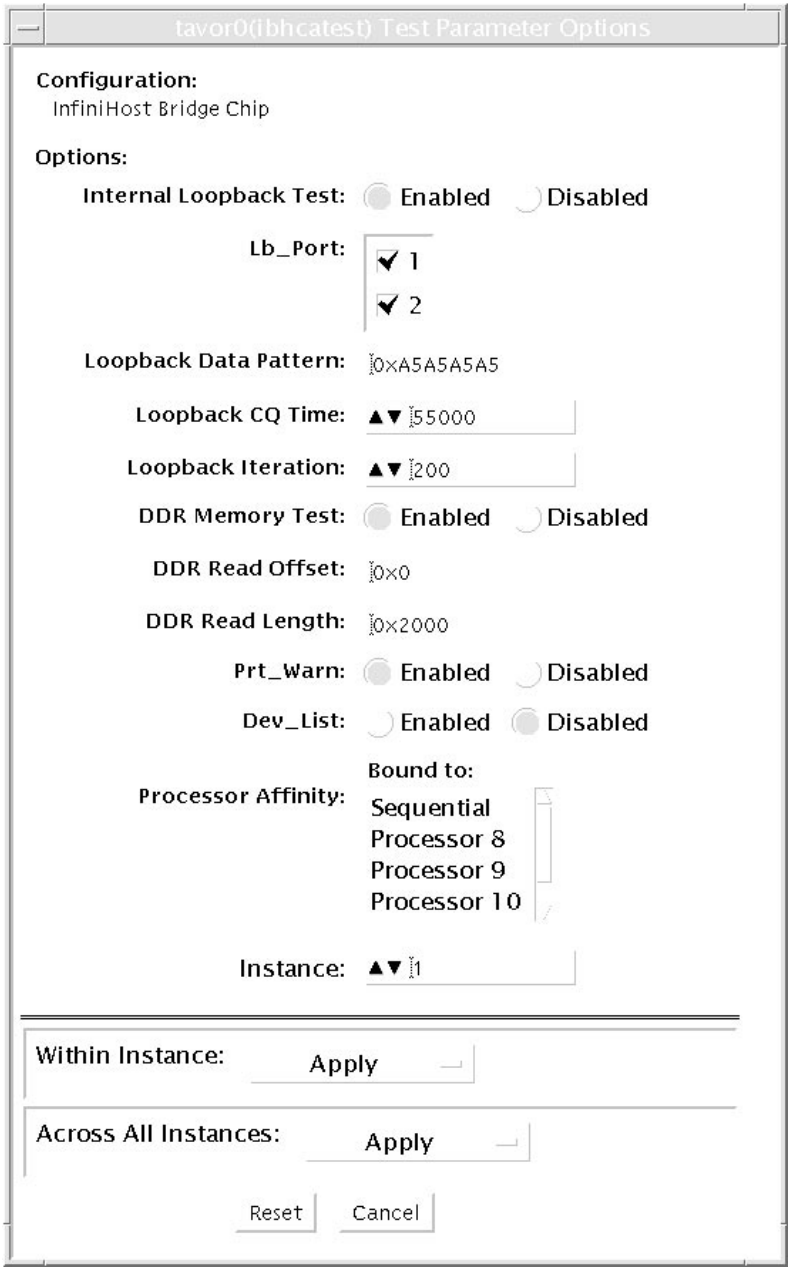


FIGURE 10-1 `ibhcatest` Test Parameter Options Dialog Box

TABLE 10-2 `ibhcatest` Options

Option	Description
`lb=`*Enabled* | *Disabled*	Turns the loopback test on or off.
`tlbport=`*1*+*2*	Loopback test on Port 1 and/or 2, default is 1+2.
`data=`*Pattern*	Specific data pattern (default patterns 0xa5a5a5a5).
`cq=`*Time*	Number of CQ polling time value (in microseconds per iteration). The default is 55000. The maximum is 1000000.
`loop=`*Number*	Number of loopback iterations for each pass. The default is 200. The maximum is 1000.
`ddr=`*Enabled* | *Disabled*	Turns the DDR memory test on or off.
`rdoffset=`*Offset*	Starts the offset of DDR memory read. The default is 0x0 in hexadecimal.
`rdsz=`*Size*	Read number of byte of DDR memory from Offset to Max address location. The default is 0x2000. The maximum is 256 MB in hexadecimal
`warn=`*Enabled* | *Disabled*	Prints a warning message when enabled.
`list`	Prints device list. No testing occurs when set.

ibhcatest Test Modes

TABLE 10-3 ibhcatest Supported Test Modes

Test Mode	Description
Connection	Queries for the Tavor firmware and hardware revision and runs the internal loopback test. The internal loopback test runs at least once depending on the duration of each pass.
Exclusive	Executes all subtests sequentially.
Functional	Executes all subtests according to what is selected.

ibhcatest Command-Line Syntax

ibhcatest [**-scruvdtlxnf**] [**-p** *n*] [**-i** *n*] [**-w** *n*] [**-o** [**dev=***text*] [**lb=***Enabled* | *Disabled*] [**tlbport=***1+2*] [**data=***Pattern*] [**cq=***Time*] [**loop=***Number*] [**ddr=***Enabled* | *Disabled*] [**rdoffset=***Offset*] [**rdsz=***Len*] [**warn=***Enabled* | *Disabled*] [**list**]]

Example:

```
# ibhcatest -p 0 -svf -o lb=Enabled, tlbport=1+2, data=0xA5A5A5A5,
cq=55000, loop=200, ddr=Enabled, rdoffset=0x0, rdsz=0x2000, warn=
Enabled, dev=tavor1
```

TABLE 10-4 ibhcatest Command-Line Syntax

Option	Description
lb=*Enabled* | *Disabled*	Turn the loopback test on or off.
tlbport=*1+2*	Loopback test on Port 1 or 2. The default is 1+2.
data=*Pattern*	Specific data pattern (default patterns 0xa5a5a5a5).
cq=*Time*	Number of CQ polling time value (in microseconds per iteration), default: 55000; max: 1000000.
loop=*Number*	Number of loopback iterations for each pass. The default is 200. The maximum is 1000.
ddr=*Enabled* | *Disabled*	Turn the DDR Memory test on or off.

TABLE 10-4 `ibhcatest` Command-Line Syntax *(Continued)*

Option	Description
`rdoffset=`*Offset*	Starts the offset of DDR memory read. Default is 0x0 in hexadecimal.
`rdsz=`*Size*	Read number of byte of DDR memory from Offset to Max address location. The default is 0x2000. The maximum is 256 MB in hexadecimal
`warn=`*Enabled* | *Disabled*	Prints a warning message when enabled.
`list`	Prints a device list. No testing occurs when set.

Note – 64-bit tests are located in the `/bin/64` directory, or the relative path in which you installed SunVTS. If a test is not present in this directory, then it might be available as a 32-bit test only. For more information, see "32-Bit and 64-Bit Tests" on page 5.

CHAPTER 11

Level 1 Data Cache Test (l1dcachetest)

l1dcachetest Description

l1dcachetest exercises the level 1 Data cache in the CPU module . The test writes, reads, and verifies access of multiple virtual addresses. The virtual addresses are chosen so that they cause targeted hits and misses in the cache. The test dynamically determines the size and organization of the cache and tunes the test accordingly to be effective on the l1dcache.

Note – l1dcachetest supports x86 platforms on the Solaris OS.

l1dcachetest provides data path testing of on-chip buses. With rapid move to deep sub-micron (DSM) designs, GHz clock frequencies, feature size process of 0.18 micron and below, l1dcachetest ensures the integrity of signals as they traverse conductors on a chip is becoming challenge. l1dcachetest subtests induce crosstalk noise in on-chip data buses by using Maximum Aggressor Fault (MAF) models.

l1dcahetest is self-scaling and adaptive, scaling the size of the system. l1dcachetest is multthreaded. Selection of CPU IDs is one of the options. But if that option is not specified, the test automatically retrieves the number of CPUs in

the system and creates that many threads of `l1dcachetest` to give coverage to the whole system at a given time. The test also determines the sizes and organization of `l1cache`.

`l1dcachetest` Options

To reach the following dialog box, right-click on the test name in the System Map and select Test Parameter Options. If you do not see this test in the System Map, you might need to expand the collapsed groups, or your system might not include the device appropriate to this test. Refer to the *SunVTS User's Guide* for more details.

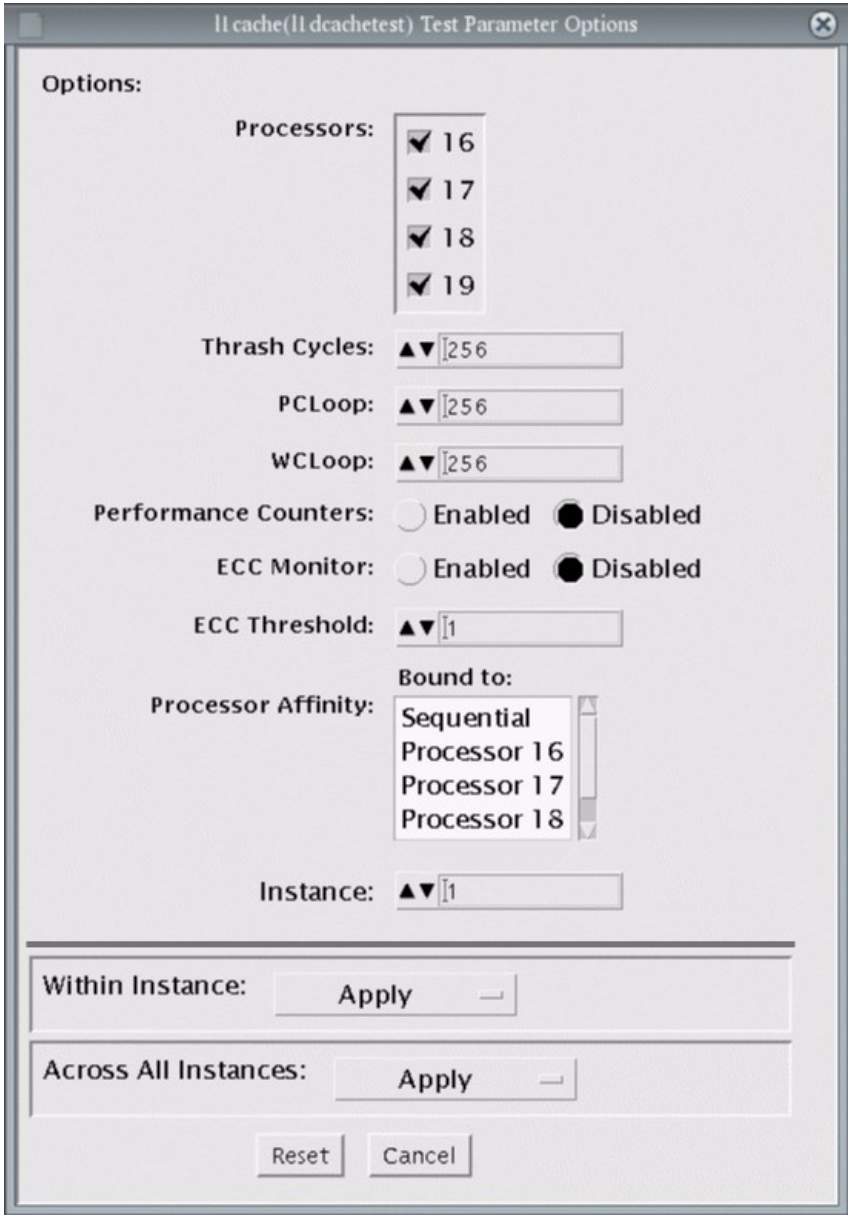


FIGURE 11-1 `l1dcachetest` Test Parameter Options Dialog Box

TABLE 11-1 `l1dcachetest` Options

Option	Description
Processors	Selects the CPU IDs for which to run this test. The test uses all CPUs on the system by default. This parameter is optional.
Thrash Cycles	Specifies the number of iterations for data cache subtests. The default is 256.
PCLoop	Specifies the number of iteration for Prefetch Cache Subtests. The default is 256. (Note – This option is displayed on system with UltraSPARC-III and UltraSPARC-IV family of processors only.
WCLoop	Specifies the number of iterations for Write Cache Subtests. The default is 256. Note: This option is displayed on systems with UltraSPARC-III and UltraSPARC-IV family of processors only.
ECC Error	Specifies whether the error messaging should be on or off. The error monitor monitors the `/var/adm/messages` file for failure messages which could be caused during test. The default is OFF.
ECC Threshold	Specifies the threshold value of the number of errors after which the test would register an error. This argument is only applicable, if the Error Monitor is on. The errors logged in the `/var/adm/messages` could be correctable errors. Ignore the errors if they are below the threshold value. The default value is 1.
Performance Counters	Enables or diables the performance counter measurements related to data cache and prefetch cache events. The default is off.

Note – Only one `l1dcachetest` gets registered for all the CPUs in the system.

Note – The `l1dcachetest` is automatically bound to a processor. Do not use the Processor Affinity option for the `l1dcachetest`.

l1dcachetest Test Modes

TABLE 11-2 l1dcachetest Supported Test Modes

Test Mode	Description
Connection	Performs the Connection subtest.
Exclusive	Performs only the l1dcachetest (full test).

l1dcachetest Command-Line Syntax

/opt/SUNWvts/bin/l1dcachetest *standard_arguments* [-scruvdtlxnf] [-p *n*] [-i *n*] [-w *n*] [-o **[M=***0+1+2+3+...* **]**,**[count**=*number* **]**, **[pcloop**=*number*], [wcloop=*number*], [**em**=*Enabled* | *Disabled*], [**threshold**=*1,255*], [**perf**=*Enabled* | *Disabled*], [**dev**=*l1cache*]]

TABLE 11-3 l1dcachetest Command-Line Syntax

Argument	Description
M=*1+2+3...*	Selects the CPU IDs for which to run this test. The test uses all CPUs on the system by default. This parameter is optional. The CPU IDs currently present in the system can be retrieved with psrinfo(1M) command. Specifying a CPU ID not present in the system or one that is offline induces an appropriate error messages from the test. For example, if you want to select CPU IDs 4, 5, 6, and 7, specify M=4+5+6+7.
count=*number*	Specifies the number of iterations for Data Cache Subtests. The default is 256.
pcloop=*number*	Specifies the number of iterations for Prefetch Cache Subtests. The default is 256. Note: This option is displayed on systems with UltraSPARC-III and UltraSPARC-IV family of processors only.
wcloop=*number*	Specifies the number of iterations for Write Cache Subtests. The default is 256. Note: This option is displayed on systems with UltraSPARC-III and UltraSPARC-IV family of processors only.
em=*Enabled* | *Disabled*	Specifies whether the error messaging should be on or off. The error monitor monitors the /var/adm/messages file for failure messages that could be caused during test. The default is off.

TABLE 11-3 `l1dcachetest` Command-Line Syntax

Argument	Description
`threshold`=*1,255*	Specifies the threshold value of the number of errors after which the test would register an error. This argument is only applicable, if the Error Monitor is on. The errors logged in the `/var/adm/messages` file could be correctable errors. Ignore the errors if they are below thrshold value. The default value is 1. If set to zero, the test still reports errors but does not stop.
`perf`=*Enabled* | *Disabled*	Enables or disables the Performance counter measurements related to Data Cache and Prefetch Cache events. The default is off.
`dev`=*l1cache*	Specifies the device to test. The default value is `l1cache`.

Note – If you do not set a value for `count` or `pcloop`, the test run with the default value of `count` and `pcloop`. To disable Data Cache subtests, specify `count=0`. For Prefetch Catch subtests, specify `pcloop=0`.

Note – 64-bit tests are located in the `sparcv9` subdirectory, `/opt/SUNWvts/bin/sparcv9/`*testname*, or the relative path to which you installed SunVTS. If a test is not present in this directory, then it might be available as a 32-bit test only. For more information, see "32-Bit and 64-Bit Tests" on page 5.

CHAPTER 12

Level 2 Cache Test (`l2sramtest`)

`l2sramtest` Description

`l2sramtest` exercises the level2 cache in the CPU module of Sun systems. In most CPUs, the level2 cache is also the external cache, but in some cases the level2 cache is on the chip. This test writes, reads, and verifies access of multiple virtual addresses. This test contains multiple subtests that try to exercise the l2cache by causing hits or misses, performing marching patterns on the l2cache cells, and writing patterns that cause electrical stress.

`l2sramtest` is self-scaling and adaptive. It scales with the size of the system. It automatically retrieves the number of CPUs in the system and internally creates that many threads of `l2sramtest` to give coverage to the whole system at a given time. For UltraSPARC-H20 and UltraSPARC-T1 systems, Level 2 cache is shared by all cores in the system thus the test retrieves the number of valid cores in the system and creates that many threads to give coverage. This test also dynamically determines the size and organization of the l2cache. You do not have to input these values.

l2sramtest Options

To reach the following dialog box, right-click on the test name in the System Map and select Test Parameter Options. If you do not see this test in the System Map, you might need to expand the collapsed groups, or your system might not include the device appropriate to this test. Refer to the *SunVTS User's Guide* for more details.

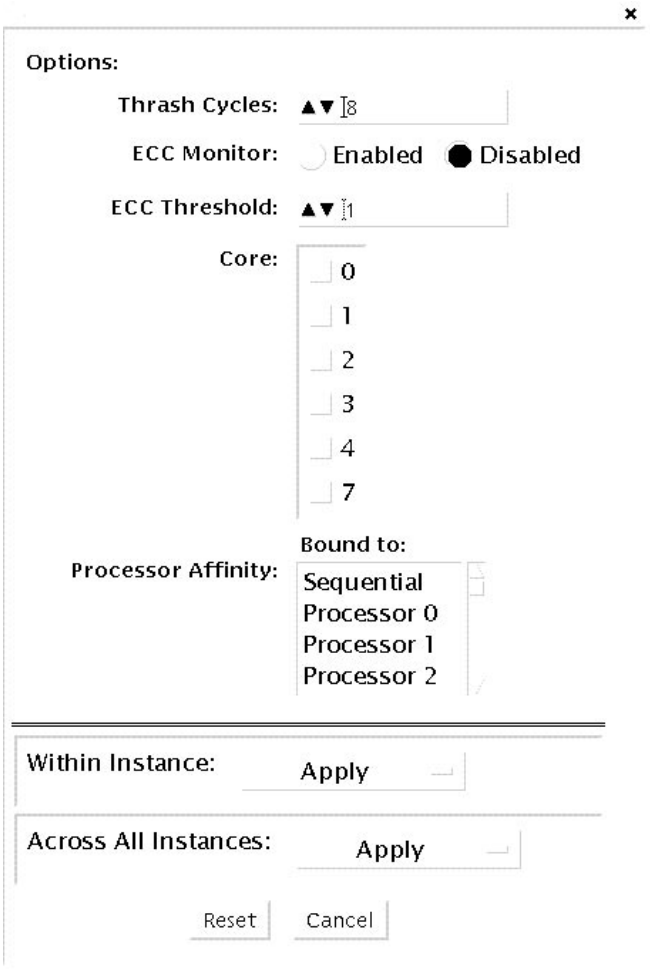


FIGURE 12-1 `l2sramtest` Test Parameter Options Dialog Box

TABLE 12-1 `l2sramtest` Options

Option	Description
Thrash Cycles	Specifies the number of thrashing cycles the test completes for the level2 cache on the system. The default value is 8.
ECC Error Monitor	Specifies whether the error monitoring should be on or off. The error monitor monitors the `/var/adm/messages` file for failure messages which could be caused due to the test. The default value is off.
Threshold	Specifies the threshold value of the number of errors after which the test would register an error. This argument is only applicable if the Error Monitor option is on. The errors that come on the `/var/adm/messages` could be correctable error, that is why the threshold value is provided for the user to give a facility to ignore the errors if they are below the threshold value. The default value is 1.
Core	This option is displayed and supported on UltraSPARC-H20 and UltraSPARC-T1 based systems only. Use the core option to perform isolation testing on selected cores only. By default, isolation testing is not enabled and system-wide testing is performed with all available cores.

Note – The `l2sramtest` automatically handles processor binding. Do not use the Processor Affinity option for the `l2sramtest`.

`l2sramtest` Test Modes

TABLE 12-2 `l2sramtest` Supported Test Modes

Test Mode	Description
Exclusive	Performs only the `l2sramtest` (full test).

`l2sramtest` Command-Line Syntax

`/opt/SUNWvts/bin/sparcv9/l2sramtest` **`-`***standard_arguments* **`-o`** [**`dev=`***l2sram*, **`count=`***[1...1023]*, **`em=`***[Enabled,Disabled]*, **`threshold=`***[0..255]*, **`core`**=[*0+1+2+...*]]

Note – The core option is supported on UltraSPARC-H20 and UltraSPARC-T1 based systems only.

Note – The `l2sramtest` is not a per CPU test. There will be only one `l2sramtest` for the whole system (one image of Solaris). It will run on all the CPUs of the domain.

TABLE 12-3 `l2sramtest` Command-Line Syntax

Argument	Description
dev=*l2sram*	Specifies the device. The default value is `l2sram`.
count=*number*	Specifies the number of thrashing cycles that the test completes for the level2 cache on the system. Default value for Offline mode is 8.
em=*Enabled/Disabled*	Specifies the enabling or disabling of the ECC Error Monitor. The default value is Disabled.
threshold=*number*	Specifies the threshold value of how many correctable ECC errors can occur in the elapsed time before `l2sramtest` reports a test failure. The default value is 1.
core=[*0+1+2+...*]	This option is supported on UltraSPARC-H20 and UltraSPARC-T1 based systems only. Use the core option to perform isolation testing on selected cores only. By default, isolation testing is not enabled and system-wide testing is performed with all available cores.

CHAPTER 13

Ethernet Loopback Test (netlbtest)

netlbtest Description

The netlbtest replaces the gemtest previously included in SunVTS. It provides functional test coverage of the devices which have device drivers that support the Ethernet loopback test. These devices include eri (the Ethernet device in the RIO chip) and ge (Gigabit Ethernet), ce (GigaSwift Ethernet), dmfe (10/100 Mbps Ethernet), and vca (Sun Crypto Accelerator 4000). The netlbtest runs in loopback (external/internal) mode.

Note – netlbtest supports x86 platforms on the Solaris OS.

Note – For x86 platforms, netlbtest only supports network drivers that have been ported over to x86 Solaris platforms. netlbtest has been tested with Sun GigaSwift-based cards on x86 systems.

The netlbtest uses DLPI RAW mode to talk to the device driver. For the purpose of this test, a packet is defined as an Ethernet header followed by the Ethernet data payload (refer to the IEEE 802.3z standard). The test generates and sends out the desired number of packets (a tunable parameter) and expects to receive the same

number of packets through the loopback interface, external or internal. If an error occurs (for example, packet mismatch or timeout), an error message indicating the type of error, its probable cause(s), and recommended action(s) is displayed on the SunVTS console.

The data sent out is generated by a random number generator, and put into a data buffer. Each time a packet is sent, it is selected from a different starting point of the data buffer, so that any two consecutively transmitted packets are not the same.

Note – Do not run `nettest` and `netlbtest` at the same time or the tests might fail.

A new debugging capability has been added in `netlbtest`. After one packet is not received, four more packets are transmitted. If all of the packets are not received within the timeout time, the test will stop with the error message, `timed out for receiving ...`. If up to four packets are missing, the test will stop with an error message, `Missed %d packet(s)...`. If a packet is received late and the current transmitted packet is not received, the test will stop with a warning message, `Packet delay...`. If the packets arrived late but within five times the timeout value and no packet is missing, the test passes.

`netlbtest` Test Requirements

You must have the Ethernet card and the device driver installed, a loopback connector in place, and Intervention mode enabled before running `netlbtest`. `netlbtest` cannot run and does not appear in the GUI if the network interface is connected to a live network. `netlbtest` also requires that the Ethernet device be configured offline before running the test. Use the `ifconfig(1M)` command to bring the Ethernet device down before running `netlbtest`. Enter the following commands to bring the interface down:

```
# ifconfig interface down
# ifconfig interface unplumb
```

To run `netlbtest`, a loopback connector must be connected to the Ethernet interface. A loopback connector provides the network interface driver the necessary link for testing, while maintaining isolation from a live network. The loopback connector is required for both internal and external tests of the Ethernet device.

The loopback cable for `ge` and Sun GigaSwift Ethernet MMF adapter (`ce` fiber) is based on the following specifications— multimode, duplex, 62.5/125 micron, sc connector, 850nm. The cable can be made by splitting a standard fiber optic cable in two. The two ends of the cable should be connected to the TX and RX ports of the adapter (the order does not matter), thus forming a loop.

The loopback connector for the `eri` device is a standard RJ-45 connector. See Appendix A in the *SunVTS User's Guide* for the diagram. The loopback connector for a Sun GigaSwift Ethernet UTP adapter (`ce` copper) is a standard RJ-45 with all 8 pins connected. See Appendix A of the *SunVTS User's Guide* for the diagram.

`netlbtest` Options

To reach the following dialog box, right-click on the test name in the System Map and select Test Parameter Options. If you do not see this test in the System Map, you might need to expand the collapsed groups, or your system might not include the device appropriate to this test. Refer to the *SunVTS User's Guide* for more details.

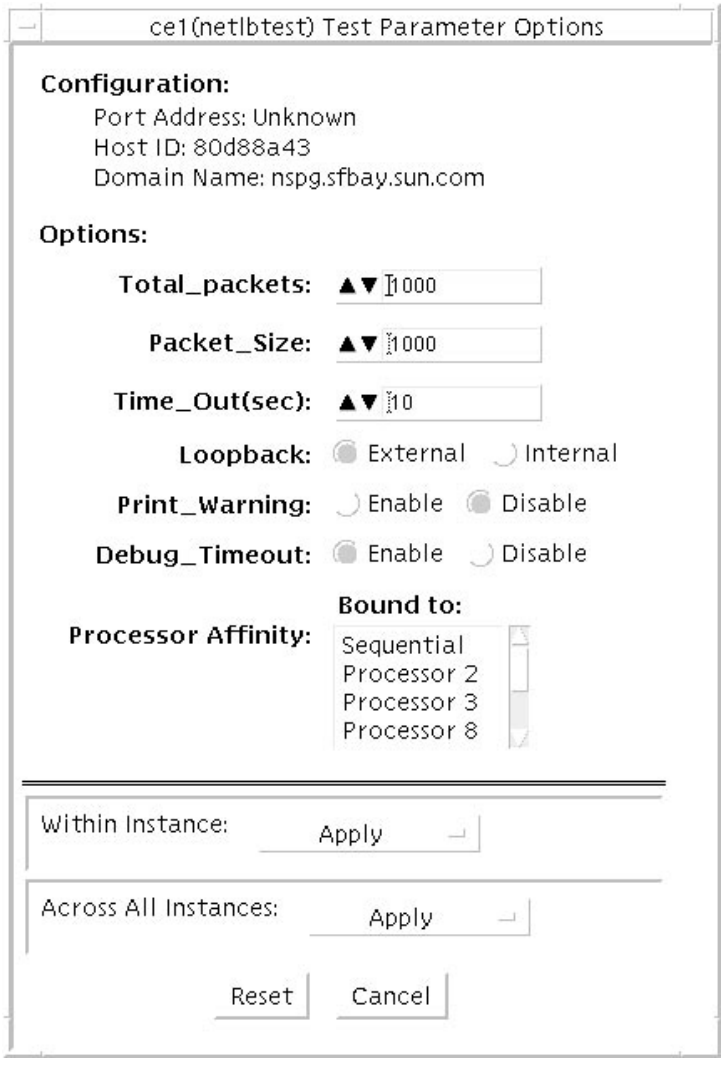


FIGURE 13-1 `netlbtest` Test Parameter Options Dialog Box

Refer to TABLE 13-1 for test parameter descriptions.

TABLE 13-1 `netlbtest` Options

`netlbtest` **Options**	**Description**
Configuration	Specifies the port address, host ID, and domain name of the system under test.
Total Packets	Specifies the total number of the packets to send. The default number of packets is 1,000. The maximum number of packets is 100,000,000.
Packet size	Determines the size (in bytes) of the packets to be transmitted. 60 <= packet size <= 1514. The default packet size is 1000 bytes.
Time_Out(sec)	Determines the amount of time (in seconds) that `netlbtest` can wait to receive packets. If no packets are received within this time frame, `netlbtest` reports an error message. The range for timeout is from 1 to 1,000 seconds.
Loopback	Determines the external and internal loopback mode. The default setting is internal loopback mode.
Print_Warning	Enables or disables the printing of warning messages. The default setting is Disable.
Processor Affinity	Binds the test to a specific processor. If no processor is specified, the test migrates between processors. This option is only available on multiprocessor systems.
Debug_Timeout	Enables or disables the debugging feature of `netlbtest`. The default setting is Disable. The range for timeout is from 1 to 1,000 seconds.

`netlbtest` Test Modes

TABLE 13-2 `netlbtest` Supported Test Modes

Test Mode	**Description**
Functional	Runs the full set of subtests. The host must not be connected to the network through the intended test device.

Since `netlbtest` requires a loopback connector, it can only be selected when Intervention mode is enabled.

`netlbtest` Command-Line Syntax

`/opt/SUNWvts/bin/netlbtest` *standard-arguments*
`-o dev=`*device*`,` **`tpkts=`***n*`,` **`pksz=`***pkt-size*`,` **`lb=`***Internal*`,` **`warn=`***Disable*`,`
`timeout=`*number-of-seconds*

TABLE 13-3 `netlbtest` Command-Line Syntax

Argument	Description
`dev=`*device-name*	Specifies the device to test such as `ge0` or `eri0`.
`tpkts=`*n*	[1...100000], count of packets to loopback. Packets number can be 1 to 100,000. The maximum number of packets is 100,000,000.
`pksz=`*pkt-size*	Determines the size (in bytes) of the packets to be transmitted. 60 <= packet size <= 1514. The default packet size is 1000 bytes.
`lb=`*Internal*	Selects internal (or external) loopback mode.
`warn=`*Disable*	Enables or disables printing of warning messages.
`timeout=`*number-of-seconds*	Determines the amount of time (in seconds) that `netlbtest` can wait to receive packets. If no packets are received within this time frame, `netlbtest` reports an error message. The range for timeout is from 1 to 1,000 seconds.

Note – 64-bit tests are located in the `sparcv9` subdirectory: `/opt/SUNWvts/bin/sparcv9/`*testname*, or the relative path to which you installed SunVTS. If a test is not present in this directory, then it might be available as a 32-bit test only. For more information, see "32-Bit and 64-Bit Tests" on page 5.

CHAPTER 14

Network Hardware Test (nettest)

nettest Description

nettest checks all the networking hardware on the system CPU board and separate networking controllers (for example, a second SBus Ethernet controller). The machine under test must be attached to a network with at least one other system on the network.

Note – nettest does support x86 platforms that use the Solaris OS. nettest supports the same set of options for both SPARC and x86 platforms. If there is a new x86 specific network interface, for which there is no entry in the nettest.conf file, then nettest will fail.

Note – This version of nettest is used for *all* networking devices, including Ethernet (ie and le), token ring (tr, trp), quad Ethernet (QED), fiber optic (fddi, nf, bf, pf), SPARCcluster™ 1 system (em), ATM (sa, ba), HiPPI, 100-Mbits per second Ethernet (be, hme), and GigaSwift Ethernet (ce) devices.

The nettest mainly uses the Internet Control Message Protocol (ICMP), and requires at least two machines on a network—the machine under test and another machine reliable enough to be a test target. Both machines must support the

Transmission Control Protocol/Internet Protocol (TCP/IP) (ICMP is part of TCP/IP). The target machine must either be configured to respond to ICMP broadcast or to RPC broadcast.

First `nettest` determines the target machine(s) to test against. If no targets are specified, it sends an ICMP broadcast to find them. If the test fails to find all necessary `nettest` targets, it tries RPC broadcast to the RPC port mapper daemon. If you specify the targets, `nettest` uses the specified target(s) instead.

After finding the necessary targets, `nettest` performs the following tests:

- Random test – Sends out 256 packets with random data length and random data.
- Incremental test – Sends out packets with length from minimum to maximum packet size using incremental data. (Minimum and maximum values differ for each device.)
- Pattern test – Sends 256 packets of maximum length, where each packet contains one test pattern, and all byte patterns (0 to 0xFF hex) are used. That is, the first packet contains pattern 0, the second packet contains pattern 1, and so on, until the last packet pattern of 0xFF.

Note – `nettest` is a scalable test. However, the maximum number of networked devices allowed on a system is 255, and the number of instances for each device is limited to 2. If you start the SunVTS exerciser using the `-i` option to specify a default number of instances for all tests, `nettest` cannot assign more than 2 instances per each networked device.

`nettest` Options

To reach the following dialog box, right-click on the test name in the System Map and select Test Parameter Options. If you do not see this test in the System Map, you might need to expand the collapsed groups, or your system might not include the device appropriate to this test. Refer to the *SunVTS User's Guide* for more details.

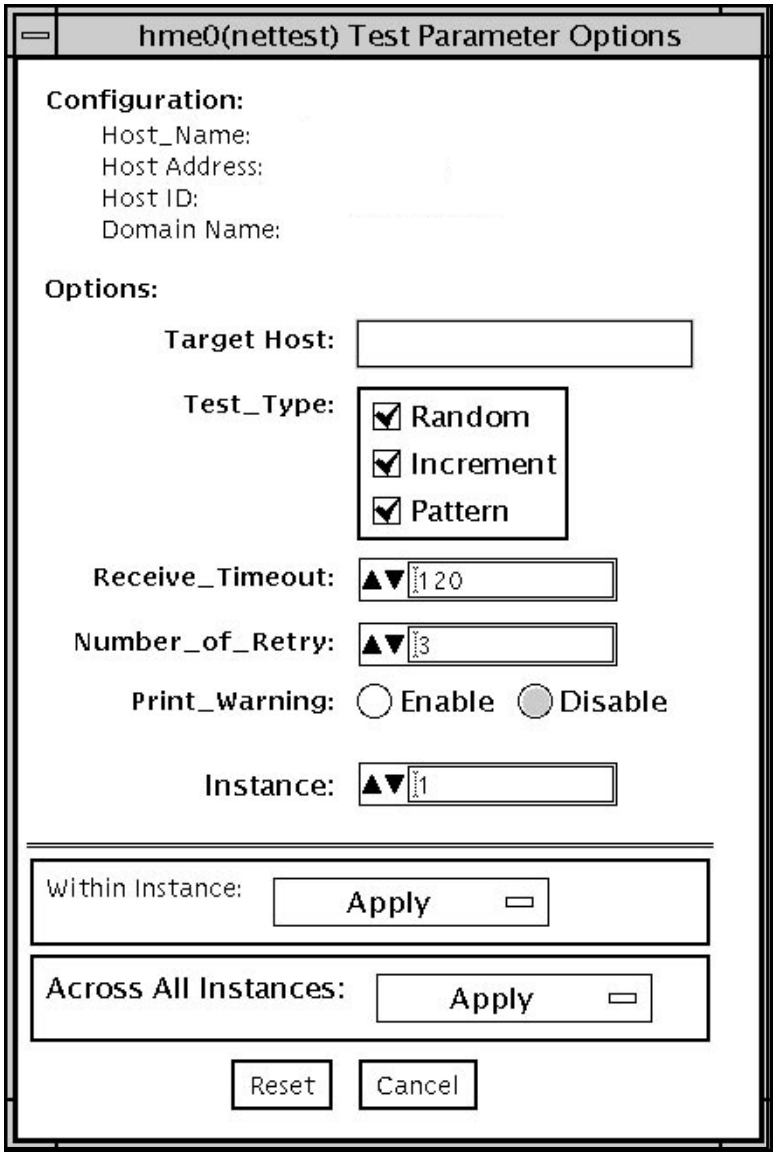


FIGURE 14-1 `nettest` Test Parameter Options Dialog Box

The Configuration section specifies the host name, host ID, host address, and domain name of the system being tested.

TABLE 14-1 `nettest` Options

`nettest` **Options**	**Description**
Target Host	Specifies one or more targets to be tested against. Target host entries can be either a host name or an Internet address. When no target host is specified, the test finds necessary targets through broadcasting. The default setting leaves this field empty.
Receive Timeout field	The default is 120 seconds, but can be changed. Use a range from 0 to 600 seconds.
Number of Retries field	The default number of retries before flagging an error is three, but can be changed. Use a range between 0 to 128 retries.
Print Warning	Disabled by default. Click Enable to see warning errors, such as retry on timeout.

`nettest` Test Modes

Both Connection and Functional test modes are supported by `nettest`. Different test schemes are performed on the network device based on the mode selected.

TABLE 14-2 `nettest` Supported Test Modes

Test Mode	**Description**
Connection	Checks whether the device is connected. It searches through all the network interfaces for a specified device name. If `nettest` does not find the device connected, the test fails; otherwise it returns: `device is connected.`
Functional	Performs all three tests (Random test, Incremental test, and Pattern test) sequentially. You may specify options that performs heavy stress testing.
Online	Performs only the Random test.

nettest Command-Line Syntax

/opt/SUNWvts/bin/nettest *standard_arguments* **-o target=***h1+h2+...***, dev=***interface***,test=***type***,packets=***n***,pattern=***hex***,timeout=***seconds***, retry=***n***,warn**

TABLE 14-3 nettest Command-Line Syntax

Argument	Description
target=*h1***+***h2***+...**	A list of test targets by host name or Internet address.
dev=*interface*	Network interface name. The default value is `le0` for Ethernet networks.
test=*type*	The test type Random, Increment, or Pattern for the desired test. The default value is Random+Increment+Pattern (all tests).
packets=*n*	Number of random/pattern packets. The default is 256.
pattern=*hex*	Specifies a data pattern, in hexadecimal form. The default is all patterns from `0` to `0xff`.
timeout=*seconds*	Indicates the number of seconds to wait before a timeout. The default is 120 seconds.
retry=*n*	Indicates the number of test timeout retries. The default is three retries.
warn	When enabled, prints warning messages.

Note – 64-bit tests are located in the `sparcv9` subdirectory: `/opt/SUNWvts/bin/sparcv9/`*testname*, or the relative path to which you installed SunVTS. If a test is not present in this directory, then it might be available as a 32-bit test only. For more information, see "32-Bit and 64-Bit Tests" on page 5.

CHAPTER 15

Physical Memory Test (pmemtest)

pmemtest Description

The pmemtest checks the physical memory of the system, and reports hard and soft error correction code (ECC) errors, memory read errors, and addressing problems. The pseudo driver mem reads the physical memory.

Note – pmemtest supports the same set of options for both SPARC and x86 platforms.

This test reads through all the available physical memory. It does not write to any physical memory location.

pmemtest Options

pmemtest is supported both in physical mapping and logical mapping displays in the UI. In physical mapping, pmemtest provides support to test the memory on a per-board basis. Users can select the pmemtest that is displayed under the physical memory board, that is to be tested and test only that board. In logical mapping, the pmemtest options apply to the complete memory across all boards.

To reach the following dialog box, right-click on the test name in the System Map and select Test Parameter Options. If you do not see this test in the System Map, you might need to expand the collapsed groups, or your system might not include the device appropriate to this test. Refer to the *SunVTS User's Guide* for more details.

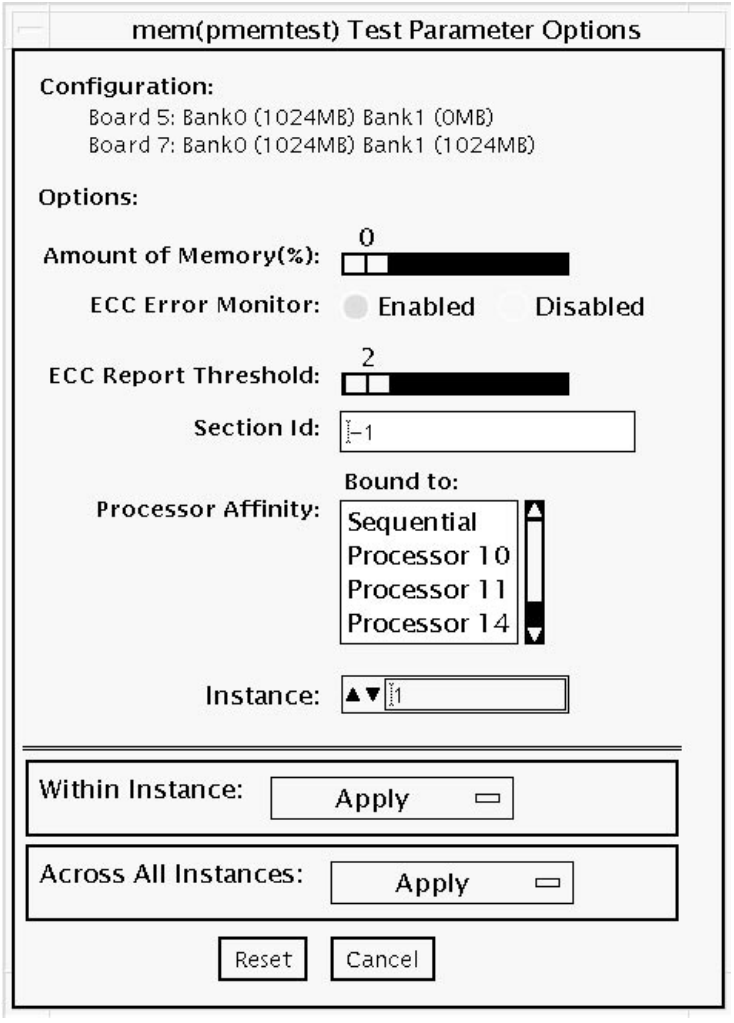


FIGURE 15-1 `pmemtest` Test Parameter Options Dialog Box

TABLE 15-1 `pmemtest` Options

`pmemtest` **Options**	**Description**
Configuration	Shows the total amount of physical memory, rounded up to the nearest MB, that is probed by the SunVTS kernel.
Amount of Memory	Specifies the percentage of the physical memory to be tested. The default 0% ensures dividing the total memory equally among instances which results in 100% coverage at the completion of every test pass. One test pass includes one pass each by all instances.
ECC Error Monitor	Enables or disables ECC error monitoring.
ECC Report Threshold	For UltraSPARC servers only. Determines how many correctable ECC errors occurred in the elapsed time before pmemtest reports a test failure. A value of zero results in no report of any correctable ECC errors. The default is 2.
Section ID	For UltraSPARC servers only. When set to -1, `pmemtest` will test one memory section in each pass, automatically testing each subsequent memory section as testing progresses. When set to a number other than -1, only the section specified is tested. A section is defined by the pass and instance number settings.
Instance	Instances are the number of copies of `pmemtest` to run simultaneously on the memory being tested.

Note – The amount of memory option is specified on a per-instance basis. The real memory coverage for one test pass depends on the amount of memory option and the number of instances. For example, if there are four instances, and each instance specifies 50% for the amount of memory option, then this will result in 200% (4 times 50%) coverage on each test pass. For guaranteed 100% memory coverage for each test pass, choose the default percentage size option as 0% for all instances.

pmemtest Test Modes

TABLE 15-2 pmemtest Supported Test Modes

Test Mode	Description
Connection	One percent of the memory is read. pmemtest also informs you how much physical memory is available. For sun4m, sun4u, and UltraSPARC servers, the test reports the ECC errors that have occurred since it was last invoked. The test reports ECC errors for a particular CPU or memory board when physical mapping is selected, otherwise it provides the SIMM number of the ECC memory error.
Functional	The amount of memory to be read can vary. By default 100% of the memory is tested. The test reports ECC errors for a particular memory board when physical mapping is selected, otherwise it provides the SIMM number of the ECC memory error.
Online	The amount of memory to be read can vary. By default 100% of the memory is tested. The test reports ECC errors for a particular memory board when physical mapping is selected, otherwise it provides the SIMM number of the ECC memory error.

pmemtest Command-Line Syntax

For 32-bit configurations:

/opt/SUNWvts/bin/pmemtest *standard-arguments* **-o size=[0-100], dev=** *device-name*, **eccmon=***Enabled* | *Disabled*, **threshold=***report-threshold*, **bdinfo=** *number*, **section=***section-id*

For 64-bit configurations:

/opt/SUNWvts/bin/sparcv9/pmemtest *standard-arguments* **-o size=[0-100], dev=***device-name*, **eccmon=***Enabled* | *Disabled*, **threshold=***report-threshold*, **bdinfo=***number*, **section=***section-id*

TABLE 15-3 `pmemtest` Command-Line Syntax

Argument	Description
`size=`*`0-100`*	Specifies the percentage of memory to be tested. The default is 0% (for 100% memory coverage).
`dev=`*device-name*	Specifies the device to test, for example, `mem`.
`eccmon=`*Enabled* | *Disabled*	Enables or disables ECC error monitoring.
`threshold=`*report-threshold*	For UltraSPARC servers only. Determines how many correctable ECC errors occur before they are reported as an error causing `pmemtest` to report a failure. A value of zero results in no report of any correctable ECC errors. The default is 2.
`bdinfo=`*number*	For UltraSPARC servers only. This argument indicates board number information. For example, if board 0 and board 5 have memory and you want the test to read the memory on both boards, then this argument should read `bdinfo`=33 (2**5+2**0). The `bdinfo` value can be specified as 0 to test the memory present on all boards.
`section=`*section-id*	For UltraSPARC servers only. When set to `-1`, `pmemtest` tests one memory section in each pass, automatically testing each subsequent memory section. When set to a number other than `-1`, only the section specified is tested. A section is defined by the pass and instance number settings.

Note – 64-bit tests are located in the `sparcv9` subdirectory: `/opt/SUNWvts/bin/sparcv9/`*testname*, or the relative path to which you installed SunVTS. If a test is not present in this directory, then it might be available as a 32-bit test only. For more information, see "32-Bit and 64-Bit Tests" on page 5.

CHAPTER 16

RAM Test (`ramtest`)

`ramtest` Description

`ramtest` is designed to stress the memory modules (RAM) instead of the whole memory subsystem. The test is optimized to achieve large memory bandwidth on UltraSPARC III (USIII) and UltraSPARC II (USII) class of CPUs. `ramtest` has an integrated ECC error monitor which reports the ECC errors found during the test run.

For x86 systems, the Exclusive mode testing puts high stress on the memory and the system interconnect.

Note – `disktest` is supported on x86 platforms with the Solaris OS.

Caution – This is an Exclusive mode test. Do not run any other application during this test.

`ramtest` Options

To reach the following dialog box, right-click on the test name in the System Map and select Test Parameter Options. If you do not see this test in the System Map, you might need to expand the collapsed groups. Refer to the *SunVTS User's Guide* for more details.

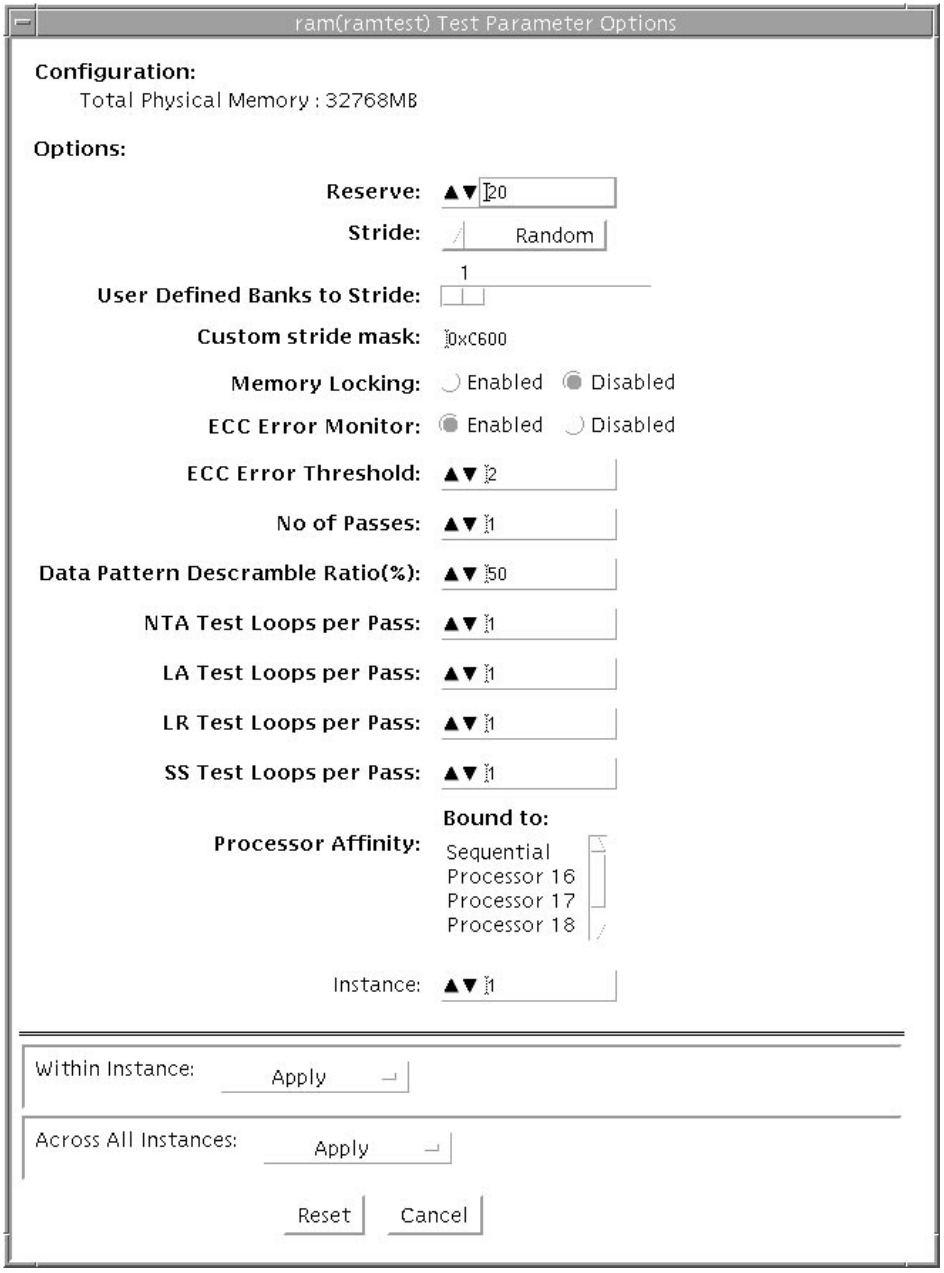


FIGURE 16-1 `ramtest` Test Parameter Options Dialog Box

The following table details the `ramtest` options:

TABLE 16-1 `ramtest` Options

`ramtest` **Options**	**Description**
Reserve	Reserve option represents the percentage of physical memory that is assumed to be in use by the OS or other processes. If you see excessive swapping while running `ramtest`, increase this percentage. The default is 20%; this means that `ramtest` allocates 80% of physical memory size for testing. Swapping decreases stress on memory and increases it on the system itself. For memory testing purposes, minimize swapping by tuning the reserve option. If the allocation or locking (in case Memory Locking is enabled) does not succeed, the amount of memory is reduced and the allocation process is repeated. Once the allocation succeeds, the amount of memory allocated is displayed in the messages.
Stride	By default this option is set to Random. It can be set to Column or Row also. In case of random, either Row or Column are randomly selected for each pass. Value of stride defines the memory locations addressed consecutively in certain subtests, in a hardware dependent manner. All testable memory is still tested. Using different strides, checks coupling among different sets of memory cells; therefore random is the recommended value for this option unless both Column and Row are being explicitly used in different instances. For FA type of uses, stride may also be set to UserDefined, in this case the test will stride the number of banks specified in the userstride option. Stride may be set to Custom in which case the stride values are randomly selected from the strides specified in the stridemask value. Stride may be set to “Custom” in which case the stride values are randomly selected from the strides specified in the “stridemask” value.
User-Defined Banks to Stride	Set the number of banks that the test should stride. The value is currently limited to between 1 and 16. Row striding is not possible while using this option.

TABLE 16-1 `ramtest` Options *(Continued)*

`ramtest` **Options**	**Description**
Stride Mask	Specifies the strides used. Each thread selects one of the stride values from stridemask by selecting one of the bits in the mask. The bits in the stridemask value represent the Least Significant Bit of the stride. Thus a value of 0x4000 calls for a stride of 16384 (using Bit 14 of the address). Mulitple bits can be set mixing row and column strides. The Memory Controller section of the PRM for the CPU of the test system for information on how the memory reference address is divided between rows and columns in the DRAM. The value can be specified as a decimal (NNN), hexadecimal (0xNNN), or octal (0NNN) value. The maximum value is 0x400000 (4194304). The default value is 0xC600 which represents strides using Bits 15, 14, 10, and 9.
Memory Locking	By default memory locking is Disabled. To turn it on, set lock to Enabled. This test uses ISM to lock the memory into the core, which allows 4 MB virtual pages and avoids swapping. Running without locking, adds more randomness to the addressing sequence.
ECC Error Monitor	ECC Monitor is Enabled by default. The ECC error monitor runs as a separate thread in the test. When an ECC error is detected, the message is displayed in the test output. Turn off the monitor by setting this option to Disabled. The ECC Monitor option is not supported on x86 platforms and an appropriate warning is displayed and the test proceeds based on other options.
ECC Error Threshold	The number of ECC errors after which the test will stop (if ECC monitor is running). When the threshold is reached, the test exits with a nonzero exit code. If set to zero, the test will still report all the errors but will not stop. The default of threshold is 2. The ECC Threshold option is not supported on x86 platforms and an appropriate warning is displayed and the test proceeds based on other options.
Number of Passes	Specifies the number of passes, in the same instance. Increase passes in case lock is enabled. This saves time spent on locking the memory every time a new process or instance is spawned by the SunVTS kernel. This pass has no relation with the system passes in the SunVTS infrastructure. It appears that `ramtest` is taking longer to complete system passes.

TABLE 16-1 `ramtest` Options *(Continued)*

`ramtest` **Options**	**Description**
NTA March Test	Specifies number of loops of NTA March(30N) Test, per pass. Increasing the number of loops of any subtest increases the relative time spent on that subtest in each pass. This increase also increases the time taken to complete a pass. NTA March Test attacks coupling and stuck at faults. NTA March is efficient at finding single-, double-, and some triple-bit errors. Depending on the stride option, coupling faults can be found between cells in adjacent columns, or rows that are targeted. Test time will be higher when row striding is selected because of greater page faults generated. For efficiency purposes, total memory is divided among available CPUs.
LA March Test	Specifies number of loops of LA March(22N) Test, per pass. Increasing the number of loops of any subtest increases the relative time spent on that subtest in each pass. This increase also increases the time taken to complete a pass. LA March Test attacks coupling and stuck-at-faults.
LR March Test	Specifies number of loops of LR march(14N) test, per pass. Increasing the number of loops of any subtest increases the relative time spent on that subtest in each pass. This increase also increases the time taken to complete a pass. LR March Test attacks coupling and stuck-at-faults.
SS March Test	Specifies number of loops of SS March(22N) Test, per pass. Increasing the number of loops of any subtest increases the relative time spent on that subtest in each pass. This increase also increases the time taken to complete a pass. SS March Test attacks simple static faults. The SS March option is not supported on x86 platforms and an appropriate warning is displayed and the test proceeds based on other options.

ramtest Test Modes

TABLE 16-2 ramtest Supported Test Modes

Test Mode	Description
Exclusive	Stresses memory modules and generates enormous amount of memory traffic.

ramtest Command-Line Syntax

/opt/SUNWvts/bin/sparcv9/ramtest *standard-arguments* **[-o**

[reserve=*Integer between 0 and 90* **]**
[stride=*Row* | *Column* | *Random* | *UserDefined* | *Custom*
[userstride=*1 - 16* **]**
[stridemask=*0x40 - 0x400000*]
[lock=*Enabled* | *Disabled* **]**
[dratio=*Integer between 0 and 100* **]**
[eccmonitor=*Enabled* | *Disabled* **]**
[threshold=*Integer i; 0 = i = MAX-INT* **]**
[pass=*32 bit integer* **]**
[ntaloops=*32 bit integer***]**
[laloops=*32 bit Integer* **]**
[lrloops=*32 bit Integer* **]**
[ssloops=*32 bit Integer*]]

TABLE 16-3 `ramtest` Command-Line Syntax

Argument	Description
`reserve`	Specifies the amount of memory that will not be allocated for testing. `reserve` represents a percentage of the total physical memory in the system. When the test starts, it probes the total memory present in the system, then tries to allocate (100 - `reserve`)% of memory. If the allocation or locking does not succeed the amount of memory is reduced before the retry. Before starting the test, the amount of memory allocated for testing is displayed. Default value for `reserve` option is 20. For UltraSPACR IIIi platforms, the default value is 25. On low memory systems, keep the reserve value higher to avoid excessive swapping. For 32-bit booted systems, the reserve value represents the percentage of 4 GB rather than the percentage of total physical memory.
`stride`	By default `stride` is set to `random`. It can be set to Column or Row also. For `random`, either Row or Column are randomly selected for each pass. The value of `stride` defines the memory locations addressed consecutively in certain subtests, in a hardware dependent manner. All testable memory is still tested. Using different `stride` checks coupling among a different set of memory cells, therefore random is the recommended value for this option unless both Column and Row are being explicitly used in different instances. For FA type of uses, stride may also be set to UserDefined, in this case the test will stride the number of banks specified in the `userstride` option. `stride` may be set to Custom in which case the `stride` values are randomly selected from the strides specified in the `stridemask` value.
`userstride`	Use this option to set number of banks the test should stride. One of the good choices could be the interleave on the suspect bank, during FA. the value is limited between 1 and 16. This also means row striding is not possible while using this option.
`stridemask`	When `stride=custom` is selected, this value specifies the strides used. Each thread selects one of the stride values from `stridemask` by selecting one of the bits in the mask. The bits in the `stridemask` value represent the Least Significant bit of the stride. Thus a value of 0x4000 calls for a stride of 16384 (using bit 14 of the address). Mulitple bits can be set by mixing row and column strides. The value can be specified as a decimal (NNN), hexadecimal (0xNNN), or octal (0NNN) value. The maximum value is 0x400000 (4194304). The default value is 0xC600 which represents strides using Bits 15, 14, 10, and 9.

TABLE 16-3 `ramtest` Command-Line Syntax

Argument	Description
`lock`	By default memory locking is disabled. To turn it on set the lock to enabled. The test uses ISM to lock the memory into the core, this gives 4 MB virtual pages and avoids swapping. Running without locking adds more randomness to the addressing sequence. On low memory systems, this option can be enabled to avoid excessive swapping. Solaris 10 users, perform the following steps: 1. Issue the following command: % **`prctl $$`** If resource controls `project.max-shm-memory` and `project.max-shm-ids` are listed in the output, proceed to the next step, otherwise follow the instructions given for Solaris 9. 2. Retrieve the default project with the following command: % **`projects -d root`** This command outputs the default project name, *project1* in this example, for the Super User. 3. Set the resource control `project.max-shm-memory` with the following command: % **`prctl -t privileged -r -n \`** **`project.max-shm-memory -v 9223372036854775807 \`** **`-i project`** *project1* For further information please refer to the *Solaris Tunable Parameters Reference Manual* applicable to your Solaris release.
`eccmonitor`	ECC Monitor is enabled by default. The ECC error monitor runs as a separate thread in the test. When an ECC error is detected, the message is displayed on to the test output. The monitor can be turned off by setting this option to disabled. The ECC Monitor option is not supported on x86 platforms. An appropriate warning is displayed and the test proceeds based on other options.
`threshold`	Number of ECC errors after which the test stops (if ECC monitor is running). When the threshold is reached the test will exit with a non zero exit code. If set to zero, the test will still report all the errors but will not stop. The default threshold is 2. The ECC Threshold option is not supported on x86 platforms and an appropriate warning is displayed and the test proceeds based on other options.
`pass`	Specifies the number of passes in the single instance. Increase `pass` if `lock` is enabled. This saves time spent on locking the memory when a new process or instance is spawned by the SunVTS kernel. This `pass` has no relation with the system passes in the SunVTS infrastructure, it will appear that `ramtest` is taking longer to complete system passes.

TABLE 16-3 `ramtest` Command-Line Syntax

Argument	Description
ntaloops	Specifies number of loops of NTA march(30N) test, per pass. Increasing the number of loops of any subtest increases the relative time spent on that subtest in each pass. This increase also increases the time taken to complete a pass. NTA march test attacks stuck-at-faults, two-cell coupling faults, and some three-cell coupling faults.
laloops	Specifies number of loops of LA march(22N) test, per pass. Increasing the number of loops of any subtest increases the relative time spent on that subtest in each pass. This increase also increases the time taken to complete a pass. LA march test attacks coupling and stuck-at-faults.
ntaloops	Specifies number of loops of NTA march test, per pass. Increasing the number of loops of any subtest increases the relative time spent on that subtest in each pass. This increase also increases the time taken to complete a pass. NTA march test attacks coupling and stuck at faults.
lrloops	Specifies number of loops of LR march(14N) test, per pass. Increasing the number of loops of any subtest increases the relative time spent on that subtest in each pass. This increase also increases the time taken to complete a pass. LR march test attacks coupling and stuck-at-faults.
dratio	Descrambles ratio tunes the algorithm used to generate data patterns in `ramtest`. A descramble ratio of 100 means that all the data patterns generated will be descrambled. If a descramble ratio is 0, the test will generate the data patterns tuned toward bus noise. Default value is 50, which means that half the data patterns are descrambled.
ssloops	Specifies number of loops of SS march(22N) test, per pass. Increasing the number of loops of any subtest increases the relative time spent on that subtest in each pass. This increase also increases the time taken to complete a pass. The SS March test attacks simple static faults. The SS March option is not supported on x86 platforms. An appropriate warning is displayed and the test proceeds based on other options.
custom	When `stride=custom` is selected, this value specifies the strides used. Each thread selects one of the stride values from `stridemask` by selecting one of the bits in the mask. The bits in the `stridemask` value represent the least significant bit of the `stride`. Thus a value of 0x4000 calls for a `stride` of 16384 (using Bit 14 of the address). Mulitple bits can be set mixing row and column strides. The value can be specified as a decimal (NNN), hexadecimal (0xNNN), or octal (0NNN) value. The maximum value is 0x400000 (4194304). The default value is 0xC600 which represents strides using Bits 15, 14, 10, and 9.

Note – 32-bit tests are located in the bin subdirectory, `/opt/SUNWvts/bin/`*testname*.

Note – ECC errors returned by `ramtest` are detected by the operating system and are logged in the `/var/adm/messages` file. Review this file for more detailed information regarding errors.

Note – 64-bit tests are located in the `/bin/64` directory, or the relative path in which you installed SunVTS. If a test is not present in this directory, then it might be available as a 32-bit test only.

CHAPTER 17

Serial Ports Test (`serialtest`)

`serialtest` Description

The `serialtest` checks the system on-board serial ports (`su[0,1]`, `zs[0,1]`, `zsh[0,1]`, `se[0,1]`, `se_hdlc[0,1]`, `asy[0,1]`), and any multiterminal interface (ALM2) boards (`mcp[0-3]`). Data is written and read in asynchronous and synchronous modes using various loopback paths. However, the x86 UART driver does not support internal loopback and synchronous mode.

Note – `serialtest` supports testing the `asy` driver (16550 UART). The 16550 device only works in asynchronous mode. The maximum baud rate is 115200.

Note – The 16550 UART can support two devices but some systems only carry one physical port (TTYS1 or TTYB, `/dev/term/b`, known as COM2 in the x86 environment). Do not try to test port a. The operating system shows two ports, but physically there is only one port.

Note – `serialtest` supports x86 platforms that use the Solaris OS. The internal loopback and synchronous modes of this test are not supported on x86 platforms. The maximum baud rate supported on x86 platforms is 115200.

Intervention mode must be enabled to run this test. This test is nonscalable.

Loopback Connectors

The loopback test requires null modem and plug connectors, which are described in Appendix A.

There are a variety of loopback paths available. The internal loopback paths do not require an external connector. Their availability depends on the device. The `zs`(7D) device has an internal path for synchronous mode and the `se`(7D) device has an internal path for asynchronous mode. The exact type of loopback connector required depends on the system I/O panel.

The loopback for the Null Modem Ato B option is a female-to-female plug. Its pin configuration is the same as the one described for the section "9-Pin to 9-Pin Port-to-Port Loopback Cable" on page 181.

The loopback for the Plug A to A option is described in the section "9-Pin Female Single-Port Loopback Plug" on page 179.

`serialtest` Synchronous Testing Software Requirements

If you have `zs`(7D) serial ports on your machine, the synchronous devices might not exist. Look in the `/dev` directory for `zsh` (where `h`=0 and/or 1). If they do not exist, use the following procedure to create them.

▼ To Create Synchronous Devices

1. **Enter the `add_drv zsh` command.**

```
# add_drv zsh
```

2. **Enter the `devfsadm` command. Go to the `/dev` directory to check if the device nodes for `zsh` have been created.**

```
# devfsadm
# cd /dev
# ls zsh
zsh
```

3. **If these steps do not work, enter the `rem_drv zsh` command and then repeat Steps 1 and 2.**

```
# rem_drv zsh
```

`serialtest` Asynchronous and Synchronous Testing

Asynchronous Testing

Note – Only asynchronous testing is available on x86 platforms.

This mode uses the asynchronous communication protocol as described in the `zs`(7D)and `se`(7D) man pages. The `termio`(7I) interface is used for configuring port characteristics.

You can select the loopback paths to use, the amount of data to transfer, and the baud rate to be used for the transfer.

The test writes and reads data through the loopback path and compares the data to the original data. The test first sends a single character. If no errors or timeouts are detected, the rest of the data is simultaneously written and read, then compared for data integrity.

Synchronous Testing

Note – Synchronous mode is not supported on x86 platforms.

This mode uses the synchronous hdlc-framing interface as described in the `zsh`(7D) and `se_hdlc`(7D) man pages. Data is written and read in checksum-protected packets.

The user can select the loopback paths to use and the clock rate.

The synchronous test runs in three phases:

1. **The first phase looks for activity on the port. If no activity is detected for at least four seconds, the test proceeds to the next phase. If activity is detected,** `serialtest` **exits with an error.**
2. **The second phase attempts to send and receive one packet. If no packets are detected after five attempts, the test exits with an error. If a packet is returned, the result is compared to the original. If the length and content of the packets do not match exactly, the test exits with an error.**
3. **The third phase attempts to send many packets through the loop. Some packet drops are to be expected especially on a heavily loaded system. The test allows a percentage of the packets to be dropped. The user can set the drop tolerance between 0 percent and 99 percent.**

   The default is 20 percent. If the system is extremely busy, the drop tolerance should be increased. Each packet is compared with its original for length and content. If a mismatch is detected, the test exits with an error.

   This mode of testing is not available for devices connected to the super I/O port. It is, however, available for all other supported I/O ports.

`serialtest` Options

To reach the following dialog box, right-click on the test name in the System Map and select Test Parameter Options. If you do not see this test in the System Map, you might need to expand the collapsed groups, or your system might not include the device appropriate to this test. Refer to the *SunVTS User's Guide* for more details.

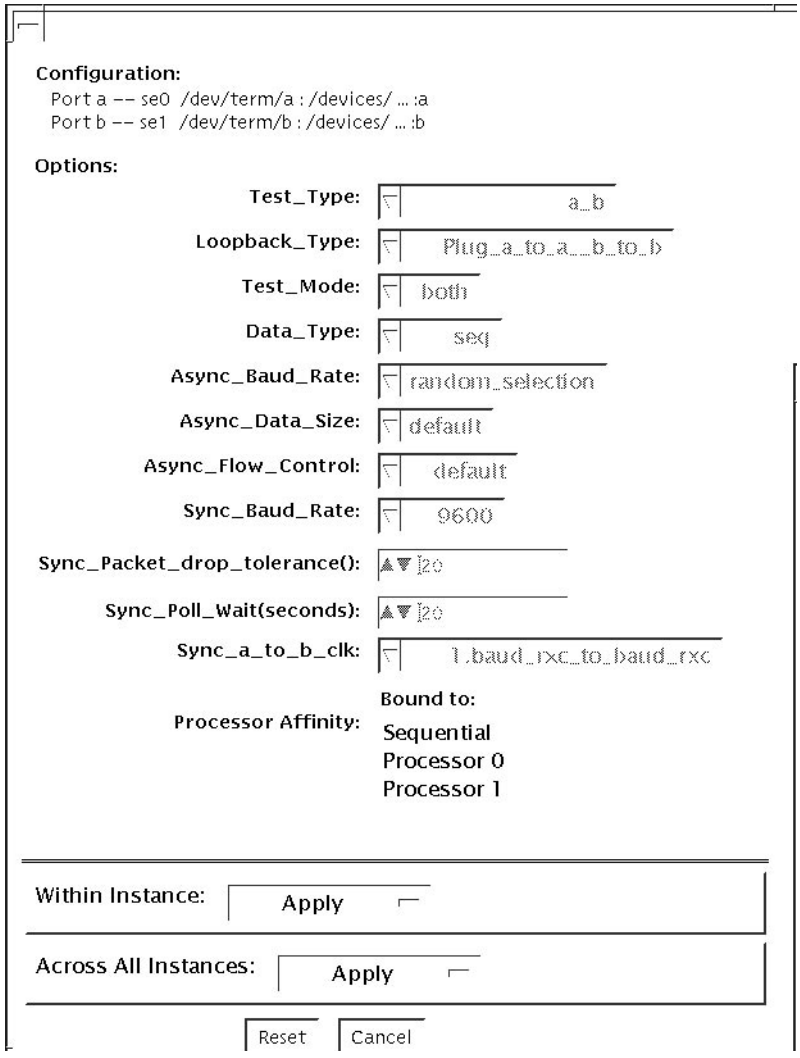


FIGURE 17-1 `serialtest` Test Parameter Options Dialog Box

TABLE 17-1 serialtest Options

Option	Description
Test Type	Selects how the test will run. Test options include: a runs the test on port A. b runs the test on port B. a_b runs the test on ports A and B sequentially. a_b_concurrent runs the test on port A and port B concurrently.
Loopback Type	Selects the loopback test. Options include: Internal is an internal path for a, b, a_b, and a_b_concurrent test types. Plug_a_to_a__b_to_b is an external loopback plug for a, b, a_b, and a_b_concurrent test types. no_modem_a_to_b is an external loopback cable for a_to_b and a_to_b_concurrent test types. Modem_a_to_b is an external loopback cable with a modem attached to generate synchronous Transmit and Receive clocks in synchronous mode. The modem a_to_b external loopback type is intended for Sun internal use only. It requires custom equipment that is not available. (Supported only for SP devices.)
Data Type	Selects the data pattern to transfer. You can select: • Random • Sequential • Alphanumeric • 0x00-0xff
Async Baud Rate	Selects the baud rate for Asynchronous mode testing. The valid rates are 110, 300, 600, 1200, 4800, 9600, 19200, 38400, 57600, 76800, 115200, 153600, 230400, 307200, 460800, and ALL. The default rate is 9600 baud. Some platforms can only support up to 38400 or 76800. If the device is connected to the super I/O port then the max baud rate supported is 115200. The test will return an error if you try to use a higher baud rate than is supported. For baud rates greater than 153600, the serial line drivers must be set for RS-423 mode and not RS-232 mode. The RS-423 and RS-232 modes are usually selected by a hardware jumper on the motherboard. Consult your hardware installation manual for more information.
Async Data Size	Selects the total number of bytes to transfer in Asynchronous mode. This can range from 1 to 10000.
Async Flow Control	Selects the type of flow control to use in Asynchronous mode testing. You can select Hardware (RTS/CTS), Software (XON/XOFF) or None. The default depends on the loopback type. Software flow control is not allowed on a, b, a_b, or a_b_concurrent loopback types.

TABLE 17-1 serialtest Options *(Continued)*

Option	Description
The following options are not available for the devices connected to the super I/O port. They are available for all other supported I/O ports.	
Test Mode	Selects the mode to run the test in. The modes available are Asynchronous, Synchronous, or Both. When Both is selected, the test runs first in Asynchronous mode then Synchronous mode.
Sync Baud Rate	Selects the device generated clock rate for Synchronous mode testing. The valid rates are from 110 to 230400. The rate does not have to be a specific value as required for async mode baud rates. The default rate is 9600. Some platforms can only support up to 38400 or 76800. The test will return an error if you try to use a higher rate than is supported. For rates greater than 100000, the serial line drivers must be set for RS-423 mode and not RS-232 mode. The RS-423 and RS-232 modes are usually selected by a hardware jumper on the motherboard. Consult your hardware installation manual for more information.
Sync Packet	Selects the tolerance level of Synchronous mode dropped packets during the many_packets subtest. The default is Drop20 percent. The valid range is from 0 percent to 99 percent. Some packet drops are expected especially at higher Toleranceclock rates and on a heavily loaded system.
Sync Poll Wait	Selects the number of seconds in additional time to wait for a Synchronous mode packet to be sent. Additional time may be needed when there is heavy system activity and time-outs are being detected. In general, the user can decrease the value to 0 when the system load is light or increase the value when there is a heavy system load.
The following option is available only for devices connected to the super I/O port. They do not apply to any other supported I/O ports.	
User Defined Baud Rate	Allows the user to set new baud rate values for the test. First select the User Defined option from the Async menu. Then enter the new value in the User Defined Baud Rate field.

serialtest Test Modes

TABLE 17-2 serialtest Supported Test Modes

Mode	Description
Connection	Attempts to open the port to determine if the device is connected. If the port does not open and if the port is not busy, the test exits with an error. If it is successful or fails with a busy or exclusive use error, then the port is considered connected, and the test passes.
Functional	Performs the selected loopback test.

serialtest Command-Line Syntax

/opt/SUNWvts/bin/sparcv9/serialtest *standard*-arguments **-o dev=** *device-name*, **porta=***port-name*, **T=***test-type*, **L=***loopback-type*, **M=***mode*, **D=***data-pattern*, **AB=***async-baud-rate*, **S=***async-data-size*, **par=***none*|*even*|*odd*, **BS=** 1|10|100|1000|3000|5000|10000, **F=***flow-control*, **B=**sync-baud-rate, **DP=** *sync-drop-tolerance*, **P=***sync-poll-wait*

Command-line syntax for devices connected to the super I/O port:

/opt/SUNWvts/bin/sparcv9/serialtest *standard-arguments* **-o dev=***device-name*, **porta=***port-name*, **T=***test-type*, **L=***loopback-type*, **D=***data-pattern*, **AB=***async-baud-rate*, **BS=***1* | *10* | *100* | *1000* | *3000* | *5000* | *10000*, **F=***flow-control*, **UAB=***User-Baud-Rate*

TABLE 17-3 `serialtest` Command-Line Syntax

Argument	Description
dev=*device-name*	Identifies the serial port(s) to test. There is no default value. You must specify a device name such as: • `se0` • `zs0`, `zs1` • `zs2`, `zs3` • `su0`
porta=*port-name*	The name of the first device of a serial device pair. The default is `A`.
T=*test-type*	Specifies the type of test to run: • a = runs the test on port A. • b = runs the test on port B. • a_b = runs the test on ports A and B sequentially. • a_b_concurrent = runs the test on port A and port B concurrently. • a_to_b = runs the test from port A to port B.
L=*loopback-type*	The type of loopback connector attached to ports: • No_modem_a_to_b • Internal_a_to_a__b_to_b • Plug_a_to_a__b_to_b • Modem_a_to_b (Sun internal only)
M=*mode*	The default test mode is asynchronous. Specify one of the following modes: • asynch • synch • both
D=*data-pattern*	Selects the data pattern to transfer. You can select: • Random • Sequential • Alphanumeric • 0x00-0xFF

TABLE 17-3 *(Continued)*`serialtest` Command-Line Syntax

Argument	Description
AB=*async-baud-rate*	Asynchronous baud rate (default = 9600). The valid values are between 110 - 460800. Note: Some platforms can only support asynchronous baud rates up to 38400 or 76800. For baud rates greater than 153600 the serial line drivers must be set for RS-423 mode and not RS-232 mode.
S=*async-data-size*	Asynchronous mode total number of bytes to write. Values range from 1 to 10000 bytes.
par=*none* | *even* | *odd*	Parity used in the async loop test. Default value is `none`.
BS=*1* | *10* | *100* | *1000* | *3000* | *5000* | *10000*	Number of bytes in each write during async loop test. Default value is 100.
F=*flow-control*	Asynchronous mode flow control: • Hardware (RTS/CTS) • Software (xon/xoff) • None
B=*sync-baud-rate*	Synchronous baud rate (default = 9600). The valid rates are between 110 - 256000. Some platforms can only support synchronous rates up to 38400 or 76800. For rates greater than 100000 the serial line drivers must be set for RS-423 mode and not RS-232 mode.
DP=*sync-drop-tolerance*	Synchronous mode drop packet tolerance (default=20 percent).
P=*sync-poll-wait*	Synchronous mode additional wait time during poll (in seconds).
UAB=*User-Baud-Rate*	Specifies any valid baud rate. Set AB to `u`.

CHAPTER 18

System Test (systest)

systest Description

The systest checks the overall functionality of a Sun system by exercising the CPU, I/O, and memory channels.

Note – systest supports x86 platforms on the Solaris OS.

Note – systest has been enhanced to enable the CPU test subtest on x86 systems. systest supports the same set of options for both SPARC and x86.

The test ensures the concurrency of the different channels by the use of Solaris threads. The test aims at stimulating failures that might be caused due to the interaction of the various different hardware modules in the system. It is very stressful on the CPU, and stresses the parallel computational capability of a multiprocessor system.

`systest` Options

To reach the following dialog box, right-click on the test name in the System Map and select Test Parameter Options. If you do not see this test in the System Map, you might need to expand the collapsed groups, or your system might not include the device appropriate to this test. Refer to the *SunVTS User's Guide* for more details.

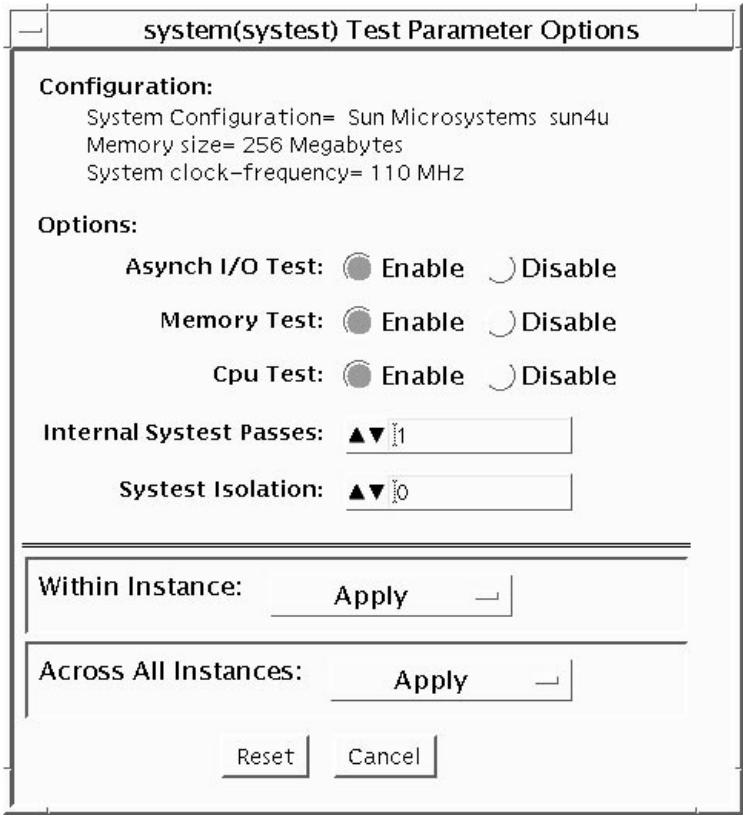


FIGURE 18-1 `systest` Test Parameter Options Dialog Box

Caution – Use discretion when defining the syspass parameter. One linpack pass (syspass=1) takes approximately 40 minutes on a server with 12 UltraSPARC® III processors. If the syspass value is set to a high value, it also increases the probability of detecting residual errors.

Caution – Use strong discretion when defining the System Isolation (`sysiso`) parameter. Be aware that `sysiso` may online / offline CPUs in the system. Do not use `sysiso` on production servers. If you choose CPUs (`sysiso=2`) isolation, the run time may be much higher than for board(s) (`sysiso=1`) isolation. The total run time for isolation can not be precisely estimated. If a residual error is found in the initial evaluation phase, the isolation functionality will online or offline CPUs in order to detect the defective boards and CPUs in the system.

Note – Users are advised to not use the Processor Affinity option for this test. Doing so reduces the effectiveness of the test.

TABLE 18-1 `systest` Options

`systest` **Options**	**Description**
Asynch I/O Test	Enables or disables the Asynch I/O subtest. The default is enable.
Memory Test	Enables or disables the Memory subtest. The default is enable.
CPU Test	Enables or disables the CPU/FPU subtests. The default is enable.
Internal System Passes	Defines the number of internal linpack passes. A set of boards and CPUs will be declared good after `syspass` number of passes. The default is 1.
System Isolation	Defines the type of isolation that `systest` needs to perform if a residual error is found in the initial evaluation phase. 0 = No isolation (default) 1 = Board(s) isolation only 2 = Board(s) and CPUs isolation

Use the default values for an initial evaluation of the system.

systest Test Modes

TABLE 18-2 systest Supported Test Modes

Test Mode	Description
Exclusive	Performs only the systest (full test).

systest Command-Line Syntax

/opt/SUNWvts/bin/systest *standard-arguments* **-o -io=***enable* | *disable* **-mem=***enable* | *disable*, **-cpu=***enable* | *disable*, **-dev=**system, **-syspass=***1,2000*, **-sysiso=***0* | *1* | *2*

TABLE 18-3 systest Command-Line Syntax

Argument	Description
io=*enable*	*disable*	Enables or disables the Asynch I/O subtest.
mem=*enable*	*disable*	Enables or disables the Memory subtest.
cpu=*enable*	*disable*	Enables or disables the CPU/FPU subtests. The CPU test is not supported on x86 platforms.
dev=*system*	Specifies the pseudo device name.
syspass=*1,2000*	Defines the number of internal linpack passes. A set of boards and CPUs is declared good after syspass number of passes. The default is 1.
sysiso=*0*	*1*	*2*	Defines the type of isolation that systest needs to perform if a residual error is found in the initial evaluation phase. 0 = No isolation 1 = Board(s) isolation only 2 = Board(s) and CPUs isolation

Note – 64-bit tests are located in the sparcv9 subdirectory: /opt/SUNWvts/bin/sparcv9/*testname*, or the relative path to which you installed SunVTS. If a test is not present in this directory, then it might be available as a 32-bit test only. For more information, see "32-Bit and 64-Bit Tests" on page 5.

Command-Line Examples

The following examples assume you execute `systest` from the command-line with verbose enabled.

Example 1 invokes the following:

```
# ./systest  -xv
```

- `systest` with default parameter values.
- I/O, MEM, and CPU subtests. The CPU test is not supported on x86 platforms.
- One internal pass of linpack and no isolation.

Example 2 invokes the following:

```
# ./systest  -xv -o io=Disable,mem=Enable,cpu=Enable,dev=system
```

- `systest` without the I/O subtest
- MEM and CPU subtests. The CPU test is not supported on x86 platforms.
- One internal pass of linpack and no isolation.

Caution – Do not perform the following `systest` examples (3 and 4) on production servers, because `systest` may online or offline CPUs.

Example 3 invokes the following:

```
# ./systest  -xv -o syspass=15,sysiso=1
```

- I/O, MEM, and CPU subtests. The CPU test is not supported on x86 platforms.
- Declares a set of boards free from residual errors after 15 internal passes of the linpack algorithm.
- If an error is found, `systest` performs boards isolation.

Example 4 invokes the following:

```
# ./systest  -xv -o syspass=10,sysiso=2
```

- I/O, MEM, and CPU subtests. The CPU test is not supported on x86 platforms.
- Declares a set of boards and CPUs free from residual errors after 10 internal passes of the linpack algorithm.

- If an error is found, `systest` performs boards AND CPUs isolation.

CHAPTER 19

Tape Drive Test (`tapetest`)

`tapetest` Description

The `tapetest` synchronous I/O test writes a pattern to a specified number of blocks (or, for a SCSI tape, writes to the end of the tape). The `tapetest` then rewinds the tape and reads and compares the data just written. The `tapetest` asynchronous I/O test sends a series of up to five asynchronous read/write requests to the tape drive, writing to the tape, and then reading and comparing the data. The terms asynchronous and synchronous referred to here, and in the Method field are not related to the SCSI messaging terms of the same name.

The `tapetest` file test writes four files to the tape and then reads them back, comparing the data. For tape library testing, the pass count is incremented only after all tapes in the library have been tested. The read/write algorithms fare enhanced for DLT tape by using a random data pattern (1.5:1 compression) and a more robust read compare algorithm. Some default parameters have also changed.

The Async I/O subtest uses the asynchronous read and write feature of the Solaris tape driver to exercise tape drives. In read-only mode the test sends a maximum of four asynchronous read packets, each with a random size and a random offset, to the tape drive. The test then waits for all outstanding I/O activity to complete before issuing another round of packets. This process continues until the whole area being tested has been covered. In read-write mode, one write packet is issued for every

four read packets to ensure a spot check of the write operation. The area of the tape to be tested is written to first in order for the test to work correctly. This test is only supported under the Solaris 8 and Solaris 9 OSs and future compatible releases.

`tapetest` Test Requirements

If you have a tape drive in your system, load a blank writable tape (scratch tape) before you start SunVTS. If you fail to do this, the `tapetest` option may display `drive type:unknown` on the option menu for the `tapetest`.

`tapetest` Options

To reach the following dialog box, right-click on the test name in the System Map and select Test Parameter Options. If you do not see this test in the System Map, you might need to expand the collapsed groups, or your system might not include the device appropriate to this test. Refer to the *SunVTS User's Guide* for more details.

`tapetest` supports 4-mm, 8-mm, DLT, 1/4-inch cartridge, and 1/2-inch front-load tape drive testing. The options available for each of the tape devices differ slightly. An example of the Test Parameter Options dialog box for a device is shown in FIGURE 19-1.

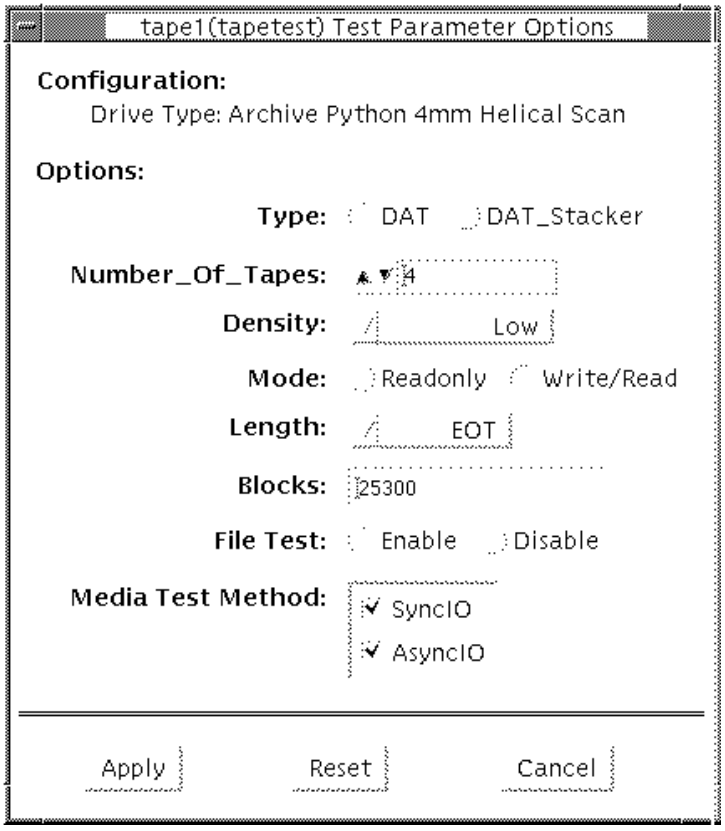


FIGURE 19-1 `tapetest` Test Parameter Options Dialog Box

Note – This test does not immediately stop after being disabled.

Note – Selecting nondefault options can significantly increase the run time for the test.

Note – The Test Parameter Options dialog box for the 1/4-inch, 1/2-inch, DLT, and 8-mm tape drives differ slightly from FIGURE 19-1.

TABLE 19-1 `tapetest` Options

`tapetest` **Options**	**Description**
Type	Normal tape drive or tape library (stacker).
Number of Tapes	The number of tapes in the tape library. `tapetest` registers a single tape library pass only after all tapes in the library pass.
Density	The following settings are available for most tape drives: • Low – Tests the `l` tape device. • Medium – Tests the `m` tape device. • Compression – Tests the `c` tape device. • All – Tests the `l`, `m`, and `c` tape devices. For half-inch tape drives, the available settings are 800, 1600, and 6250 BPI (blocks per inch). For certain QIC drives, select `QIC-11` (1-byte block ID) mode, `QIC-24` (4-byte block ID) mode, or `Both`. Note: On a DLT drive, the l and m settings both use no compression. `tapetest` does not support changing DLT capacity settings indicated on the front panel.
Mode	If you enable Write/Read mode, the test first writes to the tape and then reads it back to compare. If you enable Read_Only mode, the test assumes the tape has been properly written and merely reads and compares. This mode is useful to check proper head alignment. Note: If a read only pass is attempted and the tape was not previously written by `tapetest`, using the same test parameters currently set, a `Big Read Failure` will occur.
Length	The amount of the tape to be tested. The choices are: • EOT – The default; tests to the entire tape. • Long – The SCSI tape tests 70,000 blocks of the tape. • Short – Only the first 1000 blocks are tested. • Specified – You must type the number of blocks to be tested in the blocks field.
# of Blocks	If you select Specified under the Length option, you must type the number of blocks you want to test.

TABLE 19-1 `tapetest` Options

`tapetest` **Options**	**Description**
Blocksize	Block size specification. This option is only available for Tandberg QIC tape drives. There are two possible values. 512-bytes is for use with older tape media that have transfer size restrictions, while 64-kbytes is for use with current, high-capacity tape media. Note – This option is only available in command line interface mode. Note – With patches 110278-01 or 110211-01 applied, DLT writes either a 512 byte or 65536 byte block depending on how this parameter is set.
File Test	The tape file test sequence is as follows: 1. Writes three files. 2. Rewinds. 3. Reads part of the first file. 4. Forward spaces to the start of the second file. 5. Reads the second file. 6. Forward spaces to the start of the third file. 7. Tries to read to the end of that file for SCSI tapes only. The tape file test tries to backspace to the start of the second file and read it.
Retension	When enable is selected, the program retensions the tape.
Media Test Method	• Sync I/O—`tapetest` reads and or writes the number of blocks selected in Length. • Async I/O—`tapetest` makes four asynchronous read requests to the tape drive. If read and write testing is selected, one asynchronous write request is also sent. The test continues after completing the requests. Note – When testing Tandberg QIC drives, Async I/O testing is restricted to read-only due to asynchronous behavior differences with other tape drives. Note – This option is not associated with the synchronous data transfer request SCSI message. It is only synchronous or asynchronous in nature because the numbers of reads and writes are not synchronous to each other. The SDTR message is not invoked.

tapetest Test Modes

The tapetest supports the following test modes. It performs different test schemes on the tape device, according to the mode you select.

TABLE 19-2 tapetest Supported Test Modes

Test Mode	Description
Connection	tapetest verifies that the drive can be opened and that the drive type can be determined. If both checks are successful, or if the drive is currently busy, then the test passes. The tapetest fails if the open operation is unsuccessful for any reason other than the drive is busy.
Functional	tapetest checks the status, rewinds the tape, erases and retensions it. If the device is a cartridge tape, tapetest writes a pattern to nblks or eot (default), rewinds the tape, and then reads and compares of the pattern. On the other hand, if the device is busy or if no tape cartridge can be found in the drive, the test cannot run and fails.

tapetest Command-Line Syntax

/opt/SUNWvts/bin/tapetest *standard-arguments* **-o dev=***device-name***,s=***block-count***,d=***density***,m=***mode***,l=***length***,method=***method***,ft=***Enables* | *Disables***,ret=***Enables* | *Disables***,dat=***dat-type***,8mm=***8mm-type***,num=***magazine-size***,blocksize=***block-size*

TABLE 19-3 tapetest Command-Line Syntax

Argument	Explanation
dev=*device-name*	Specifies the *device-name* of the tape drive (required).
s=*block-count*	Specifies the number of blocks to be tested.
d=*density*	Specifies the density of the tape to open.
m=*mode*	Enables either the Write-Read or Read-Only tests.
l=*length*	Specifies the length of the test (EOT, Specified, Long, or Short).
method=*method*	Specifies the media test method (SyncI/O and or AsyncI/O) used. Note: This option does not invoke the SCSI message synchronous data transfer request. It is only asynchronous or synchronous in nature.

TABLE 19-3 `tapetest` Command-Line Syntax *(Continued)*

Argument	Explanation *(Continued)*
`ft`=*Enables* | *Disables*	Enables or disables the File test.
`ret`=*Enables* | *Disables*	Enable or disables tape retension.
`dat=`*dat-type*	If you are testing a digital audio tape drive, specify whether it is a regular DAT drive or a DAT stacker. The choices are `DAT` and `DAT_Stacker`.
`8mm=`*8mm-type*	If you are testing an 8-mm tape drive, specify whether it is a regular 8-mm tape drive or a tape library. The command line choices are `8mm` and `8mm_Library`.
`num=`*magazine-size*	If you are testing a tape library, specify the magazine size.
`blocksize=` *block-size*	This option is only available on a Tandberg QIC drive and DLT drives. Specify whether to use a 64 kbyte block transfer or a 512 byte block transfer. Use 512 bytes when testing older media in the drive. DLT supports 512 byte and 65536 byte modes

Note – 64-bit tests are located in the `sparcv9` subdirectory: `/opt/SUNWvts/bin/sparcv9/`*testname*, or the relative path to which you installed SunVTS. If a test is not present in this directory, then it might be available as a 32-bit test only. For more information, see "32-Bit and 64-Bit Tests" on page 5.

CHAPTER 20

USB Device Test (`usbtest`)

`usbtest` Description

The `usbtest` verifies the proper functioning of the hardware of the USB subsystem. The test supports audio, keyboard, and printer devices. The `usbtest` for audio verifies the proper functioning of the hardware and software components of the USB audio subsystem. Specifically, the test tests the USB audio devices, USB microphones and speakers.

Note – `usbtest` tests the USB ports using various USB devices such as the USB key board, the USB printer, or the USB audio devices. `usbtest` supports the same set of options for both SPARC and x86.

For keyboards, the test verifies whether the keyboard(s) attached to the USB bus are USB-compliant. The test flashes the LEDs of a compliant keyboard. The `usbtest` for printers verifies the proper functioning and integrity of the USB parallel port devices. The test data, either ASCII or a PostScript file, is sent to the printer through the USB bulk transfer. The test saves and restores the printer settings.

The `usbtest` options and subtest available depends on the peripheral device attached to the USB port.

usbtest Subtests

XXX lists the usbtest subtests for both the USB audio and printer devices.

TABLE 20-1 usbtest Subtests

Subtest	Description
USB Audio Devices	
Tone	Supported in Connection test mode only. This subtest generates two seconds of sound output to the speakers. You must decide, by listening, if the test passed or failed.
Record/Play	Records and plays one second of random data at 8kHz sampling. This subtest reads random data from the USB microphone port and plays the music back to the USB speakers.
Audio	Plays a 30 second music file that is output to the speakers. You must decide, by listening, if the test passed or failed. Things to listen for are distortion or lack of music.
USB Printer Devices	
getdevid	Retrieves the IEEE 1284 ID string of the printer.
Printer	Prints strings of ASCII characters (from 0x32 to 0x7e), and can also print the postscript file usbpppdata.ps to the printer.

usbtest Options

To reach the following dialog box below, right-click on the test name in the System Map and select Test Parameter Options. If you do not see this test in the System Map, you might need to expand the collapsed groups, or your system might not include the device appropriate to this test. Refer to the *SunVTS User's Guide* for more details.

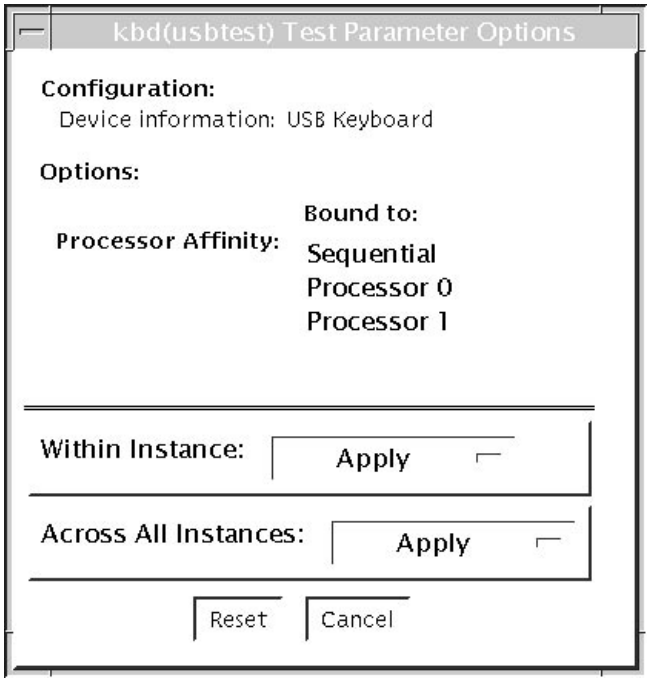


FIGURE 20-1 `usbtest` Test Parameter Options Dialog Box

TABLE 20-2 `usbtest` Opions

Option	Description
USB Audio Devices	
Volume	Specifies the volume of 0 to 255. The default is 80.
Audiotest	Enables or disables the audio music test. The default is Enable.
USB Printer Devices	
DeviceID	Specifies the path to the printer
Printer	Enables or disables the Printer subtest. This test requires a printer to be attached to the USB port.
Printer_data_type	Specifies the test data: ascii, postscript, or both.
Printer_delay_seconds	Specifies, in seconds, the delay after the Printer subtest is performed

usbtest Test Modes

This test supports Connection and Functional test modes.

TABLE 20-3 usbtest Supported Test Modes

Test Mode	Description
Connection	For a USB audio device, the test outputs two seconds of sound to the speakers For a USB keyboard device, the test runs the full test. For USB printer device, the test runs only the getdevid subtest.
Functional	For a USB audio device, the test performs Records and Plays subtest by default. You can enable Audio Music test. For a USB keyboard device, the test runs the full test. For a USB printer device, the `getdevid` subtest is enabled by default and the Printer subtest is disabled by default.

usbtest Command Line Syntax

/opt/SUNWvts/bin/sparcv9/usbtest -cvf -o dev=*dev-name*, **getdevid=***Enable* | *Disable*, **printer=***Enable* | *Disable*, **data=***ascii*, **postscript**, **delay=***0-86400*, **M=***Enable* | *Disable*, **O=***speaker* | *headphone* | *line-out*, **V=***0-255*, **MF=***musicfilename*

TABLE 20-4 usbtest Command-Line Syntax

Argument	Description
dev=*dev-name*	Target device name. Thie test runs on the specified device name. The default device name is `/dev/kbd`.
USB Printer Devices	
getdevid=*Enable* | *Disable*	Enables or disables getdevid subtest. This option is available only for a USB printer device. This option is enabled by default.
printer=*Enable* | *Disable*	Enables or disables the printer test. This option is disabled by default. This option requires a printer to be attached to the USB port.
data=*ascii*	Selects whether to send ASCII text or PostScript data to the printer.

TABLE 20-4 `usbtest` Command-Line Syntax *(Continued)*

Argument	Description
`delay=`*0-86400*	Enables you to choose a delay between passes of the printer test.
USB Audio Devices	
`M=`*Enable* | *Disable*	Enables or disables the music play test. The default is Enable.
`O=`*speaker* | *headphone* | *line-out*	Selects the output device that `usbtest` requires.
`V=`*0-255*	Music output volume. The default is 80.
`MF=`*musicfilename*	Selects the music file. The default is `music.au`.

CHAPTER 21

Virtual Memory Test (vmemtest)

vmemtest Description

The vmemtest checks virtual memory—the combination of physical memory and the swap partitions of the disk(s).

Note – This test might not stop immediately after being disabled.

Note – vmemtest supports the same set of options for both SPARC and x86 platforms.

This test uses the Solaris valloc (page-aligned) system call to allocate, write, read, and compare virtual memory. These operations normally cause heavy paging activity on the system and simulate a stressful environment for the operating system. This test also detects ECC parity errors, memory read errors, and addressing problems, and displays the corresponding virtual memory addresses on failure.

Note – Do not run the vmemtest with fwcamtest at the same time on any Sun Blade system. This will cause vmemtest to fail.

`vmemtest` Swap Space Requirements

Running this test places a significant burden on the operating system, since it uses the majority of swap space available for testing. Use the `vmemtest` swap space `reserve` option when nonSunVTS test processes are started after SunVTS testing has started. Refer to the *SunVTS User's Guide* for a complete discussion of swap space requirements.

`vmemtest` Options

To reach the following dialog box, right-click on the test name in the System Map and select Test Parameter Options. If you do not see this test in the System Map, you might need to expand the collapsed groups, or your system might not include the device appropriate to this test. Refer to the *SunVTS User's Guide* for more details.

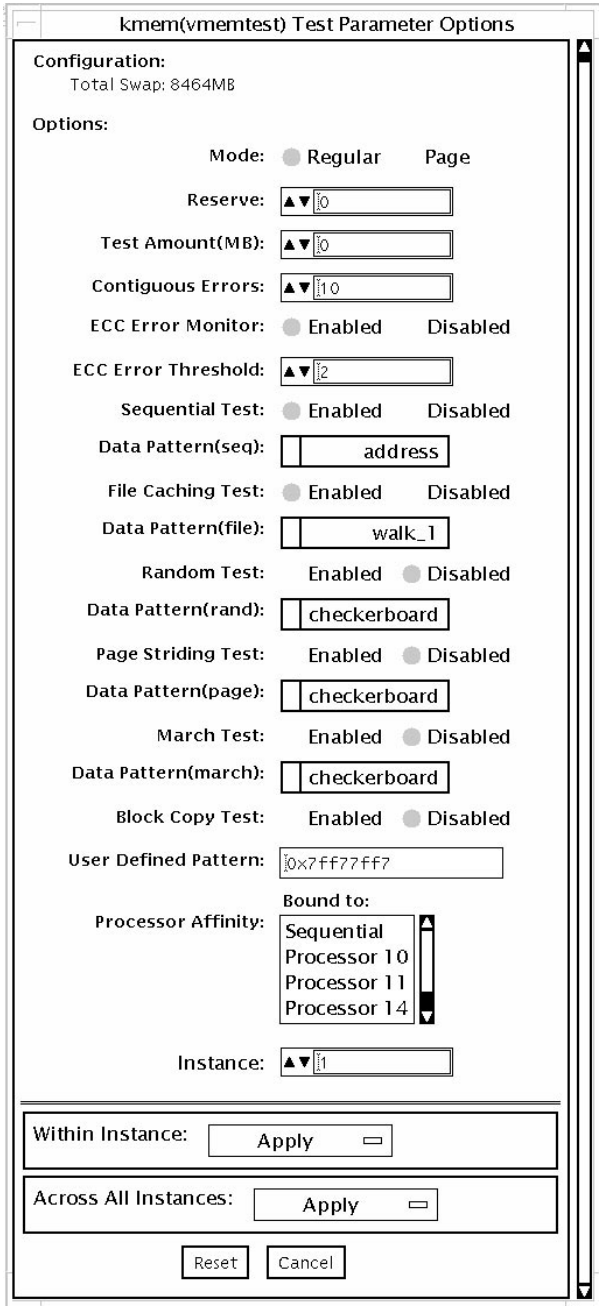


FIGURE 21-1 `vmemtest` Test Parameter Options Dialog Box

TABLE 21-1 `vmemtest` Options

vmemtest Options	Description
Mode	Two modes are available: • Regular mode tests the specified amount of memory as one block and passes as the size argument to the different test algorithm functions (subtests). • Page mode tests assign virtual memory one page at a time. Each page is mapped to the temporary file `/tmp/vmem.page` and is then paged out to storage once test data is written. Next, the temporary page is paged back into memory for a read and compare. `vmemtest` runs in Regular mode as default setting.
Reserve	Specifies the amount of memory to reserve from being tested by `vmemtest`. The test ensures that this amount of memory is left free on the system while evaluating the size of memory for testing. If specified value of reserve is zero, this test will use a default value evaluated based on the available free swap space for the instance. Trying to reserve more memory than available free memory by this instance will cause the test to fail.
Test Amount	An amount can be specified to test the virtual memory, instead of the default.The default value is 0, which means the default memory size is evaluated within the test. Record the memory configuration details on the target system before selecting nondefault setting for the amount option. If negative values are specified, the test assumes the default setting. The actual size of memory tested by the instance is always evaluated with reference to the available free swap space on the system.
`vmemtest` Configuration	The amount of memory listed in the Configuration field is equivalent to the sum of the used and available swap space amounts returned by the `swap -s` command. This number ndicates the amount of virtual memory found, rounded up to the nearest Kbyte.
Contiguous Errors	Specifies the max number of contiguous memory errors, which will be considered and counted as one noncontiguous error. The default value is 10.
ECC Error monitoring	Enables or disables ECC error monitoring.
ECC Error threshold	Determines how many correctable ECC errors occurred in the elapsed time before `vmemtest` reports a test failure. The default threshold value is 2.

TABLE 21-1 `vmemtest` Options *(Continued)*

vmemtest Options	Description
Test Method	`vmemtest` runs the Sequential and File Caching subtests by default. • Sequential subtest – Tests the whole memory from the beginning address to the end address in a sequence. • Address Random subtest – Randomly selects memory addresses with in the specified range to test. • Page Striding subtest – Noncontiguous memory test, implemented sequentially and non-sequentially. • Sequential striding – Tests from the first page to the last page, withn a specified test range. Only one word is tested per page. • Nonsequential striding – Tests randomly from first to last page, within a specified memory range. Alternates testing one word per page until all pages are tested. • Block Copy test – Writes and reads data between two memory blocks. Each memory block is half the memory to be tested. • File Caching test – Aimed at improving performance through the use of file caching in the Solaris kernel. This test is useful for large memory configurations. This test takes 30 to 70% less time than the Sequential test method.
Predefined Pattern	Select one of the following patterns to use for the test: • Address–uses the virtual addresses of the tested memory locations. • walk_1–uses a pattern that starts with 0x80000000 through 0x11111111. • walk_0–uses a pattern that starts with 0x7fffffff through 0x00000000. • 0x00000000–uses all ones and zeros for testing. • 0x5aa55aa5–uses 0x5aa55aa5 pattern. • 0xdb6db6db–uses 0xdb6db6db pattern. • Checkerboard–uses 0xaaaaaaaa patterns. • UserDefined–uses the pattern that is specified in the User Defined Pattern area (see below).
User Defined Pattern	Only used if the Predefined Pattern is set to UserDefined. The pattern specified should be in the form of an 8-digit, hexadecimal number such as 0x2a341234. Default setting is up= 0x7ff77ff7
Instance	Specifies how many copies of the `vmemtest` test to run.

vmemtest Test Modes

TABLE 21-2 vmemtest Supported Test Modes

Test Mode	Description
Functional	Writes a pattern to an amount of the specified virtual memory. The data is then read back and compared. If there is a miscompare, the virtual address is reported and the data is read again and compared.

In Functional test mode, vmemtest writes a pattern to an amount of virtual memory specified by the user. Then the data is read back and compared. If there is a miscompare, the data is read again and compared. Whenever there is a miscompare, the virtual address is reported.

vmemtest Command-Line Syntax

/opt/SUNWvts/bin/vmemtest *standard_arguments* **-o mode=***type*, **reserve=** *n*, **amount=***n*, **cerr=***n*, **eccmon=***enable* | *disable*, **eccthreshold=***n*, **type1=** *enable* | *disable*, **pp1=***pattern*, **type2=***enable* | *disable*, **pp2=***pattern*, **type3=***n*, **pp3=** *pattern*, **type4=***enable* | *disable*, **pp4=***pattern*, **type5=***enable* | *disable*, **pp5=***pattern*, **type6=***enable* | *disable*, **up=***hex_pattern*

TABLE 21-3 vmemtest Command-Line Syntax

Argument	Explanation
mode=*Page* | *Regular*	Specifies which mode of the vmemtest to run. • Page – tells the write/read memory test to proceed one system memory page at a time. • Regular – uses the valloc option to allocate the entire assigned memory, which is read and compared one long word at a time.
reserve=*n*	Specifies the amount of MB of virtual memory to reserve.
amount=*n*	Specifies the number of MB of memory to be tested instead of the default.
cerr=*n*	Specifies the maximum number of contiguous errors to be counted as one non contiguous error.
eccmon=*enabled* | *disabled*	Enables or disables the ECC error monitor.

TABLE 21-3 `vmemtest` Command-Line Syntax

Argument	*(Continued)*Explanation
eccthreshold=*n*	Specifies how many correctable ECC errors can occur in the elapsed time before `vmemtest` reports a test failure.

type1=*value*
pp1=*pattern*
	`type1` is the sequential test. The value is Enabled or Disabled. The default is Enabled. The default for the `pp1` pattern is `address`. Select the `pp1` pattern from:
	`address,walk_0,walk_1,Checkerboard,`
	`0x00000000,0xffffffff,0x5aa55aa5,`
	`0xdb6db6db,random,`*UserDefined*

type2=*value*
pp1=*pattern*
	`type2` is the File cache test. The value is Enabled or Disabled. The default is Enabled. The default for the `pp1` pattern is `address`. Select the `pp1` pattern from:
	`address,walk_0,walk_1,Checkerboard,`
	`0x00000000,0xffffffff,0x5aa55aa5,0xdb6db6db,`
	`random,`*UserDefined*

type3=*value* **pp3=***pattern*	`type3` is the Random address test. The value is Enabled or Disabled. The default is Disabled. The default of the `pp3` pattern is `checkerboard`. Select the `pp3` pattern from: `Checkerboard,0x00000000,0xffffffff,` `0x5aa55aa5,0xdb6db6db,`*UserDefined*
type4=*value* **pp4=***pattern*	`type4` is the `page_striding` test. The value is Enabled or Disabled. The default is Disabled. The default of the `pp4` pattern is `checkerboard`. Select the `pp4` pattern from: `Checkerboard,0x00000000,0xffffffff,` `0x5aa55aa5,0xdb6db6db,`*UserDefined*
type5=*value* **pp5=***pattern*	`type5` is the `march_c` test. The value is Enabled or Disabled. The default is Disabled. The default for the `pp5` pattern is `checkerboard`. Select the `pp5` pattern from: `Checkerboard,0x00000000,0xffffffff,` `0x5aa55aa5,0xdb6db6db,`*UserDefined*
type6=*value*	`type6` is the `Block_Copy` test. The value is Enabled or Disabled. The default is Disabled. Note – The `Block_Copy` subtest uses its own set of the data patterns predefined in the test. It does not require any user specified data patterns for testing.
up=*hex_address*	Only used if the `pp` argument is set to `UserDefined`. The pattern specified should be in the form of a 8-digit, hexadecimal number such as `0x2a341234`.

Note – 64-bit tests are located in the `sparcv9` subdirectory: `/opt/SUNWvts/bin/sparcv9/`*testname*, or the relative path to which you installed SunVTS. If a test is not present in this directory, then it might be available as a 32-bit test only. For more information, see "32-Bit and 64-Bit Tests" on page 5.

APPENDIX **A**

Loopback Connectors

Loopback Connection Overview

Loopback connectors aid in testing communication ports. The connectors take the form of either a single plug or a port-to-port cable with some communication connections shorted (looped-back).

Note – Loopback connectors must be wired properly and connected firmly for the Serial Port tests to work correctly. Miswired, poorly soldered, or missing loopback connectors can cause erroneous diagnostic error messages.

The following table depicts the pin assignments for most loopback plugs and cables that may be used when testing a system.

TABLE A-1 Loopback Connector Pin Assignments

Signal Description	EIA	CCITT #	RS-449 A	RS-449 B	DIN 8 8-pin round	DB9 9-pin	DB25 25-pin	Direction	Alpha ID
Chassis/Frame Ground	AA	101	1	NC*	NC*	NC*	1	None	AA
Transmit Data (TxDa)	BA	103	4	22	3	3	2	Output	BA
Receive Data (RxDa)	BB	104	6	24	5	2	3	Input	BB
Request To Send (RTSa)	CA	105	7	25	6	7	4	Output	CA
Clear To Send (CTSa)	CB	106	9	27	2	8	5	Input	CB
Data Set Ready (DSRa)	CC	107	11	29	NC*	6	6	Input/ output	CC
Signal Ground (SG)	AB	102	19	NC*	4	5	7	None	AB
Data Carrier Detect (DCDa)	CF	109	13	31	7	1	8	Input	CF
Transmit Clock In (TRxCa)	DB	114	5	23	NC*	NC*	15	Input	DB
Receive Clock In (RTxCa)	DD	115	8	26	8	NC*	17	Input	DD
Data Terminal Ready (DTRa)	CD	108	12	30	1	4	20	Output	CD
External Clock Out (TRxCa)	DA	113	17	35	NC*	NC*	24	Output	DA
Secondary Data Carrier Detect (DCDb)	SCF	122	NC*	NC*	NC*	NC*	12	Input	SCF
Secondary Clear to Send (CTSb)	SCB	121	NC*	NC*	NC*	NC*	13	Input	SCB

TABLE A-1 Loopback Connector Pin Assignments *(Continued)*

Signal Description	EIA	CCITT #	RS-449 A	RS-449 B	DIN 8 8-pin round	DB9 9-pin	DB25 25-pin	Direction	Alpha ID
Secondary Transmit Data (TxDb)	SBA	118	NC*	NC*	NC*	NC*	14	Output	SBA
Secondary Receive Data (RxDb)	SBB	119	NC*	NC*	NC*	NC*	16	Input	SBB
Secondary Request to Send (RTSb)	SCA	120	NC*	NC*	NC*	NC*	19	Output	SCA

*NC = No connection

25-Pin RS-232 Loopback Plug

FIGURE A-1 shows the RS-232 and RS-423 single-port loopback plug, which is a specially wired male DB-25 connector. The plug is connected to a serial port in the back of the system under test. The following table lists the pin connections for connecting the first plug to the second plug:

TABLE A-2 25-pin RS-232 Loopback Plug Wiring Plug Connections

First Plug	Second Plug
Pin 3	Pin 2
Pins 6 and 8	Pin 20
Pins 15 and 17	Pin 24

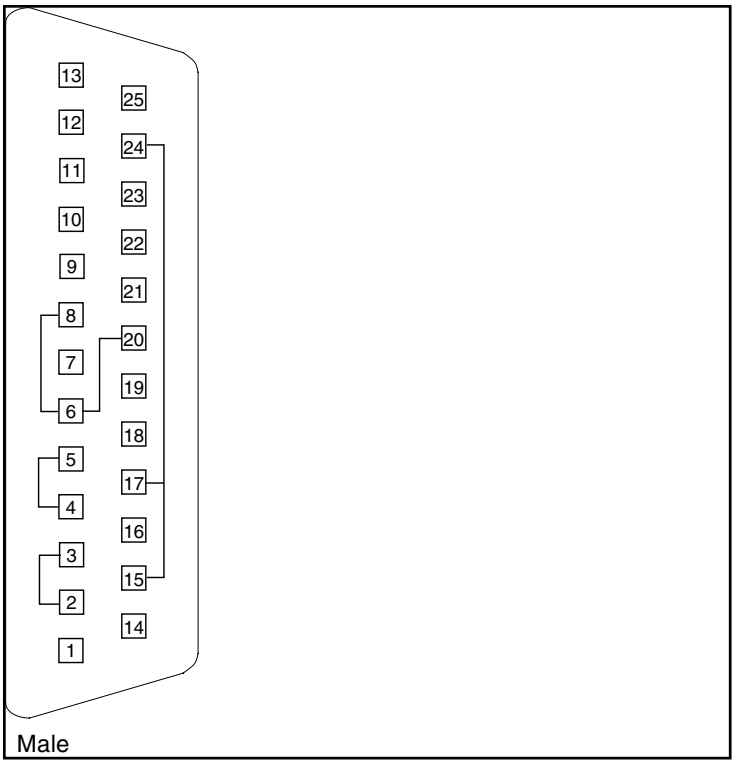


FIGURE A-1 25-pin RS-232 Loopback Plug Wiring Diagram

25-Pin RS-232 Port-to-Port Loopback Cable

FIGURE A-2 shows how to connect 25-pin RS-232 and RS-423 port to 25-pin RS 232 and RS 423 port loopback cables (two DB-25 connections). The cable is connected to a pair of serial ports in the back of the system under test. Both plugs are male. The following table lists the pin connections for connecting the first plug to the second plug:

TABLE A-3 25-pin RS-232 Port-to-Port Loopback Cable Wiring Plug Connections

First Plug	Second Plug
Pin 2	Pin 3
Pin 3	Pin 2
Pin 4	Pin 5
Pin 5	Pin 4
Pins 6 and 8	Pin 20
Pin 7	Pin 7
Pins 15 and 17	Pin 24
Pin 20	Pins 6 and 8
Pin 24	Pins 15 and 17

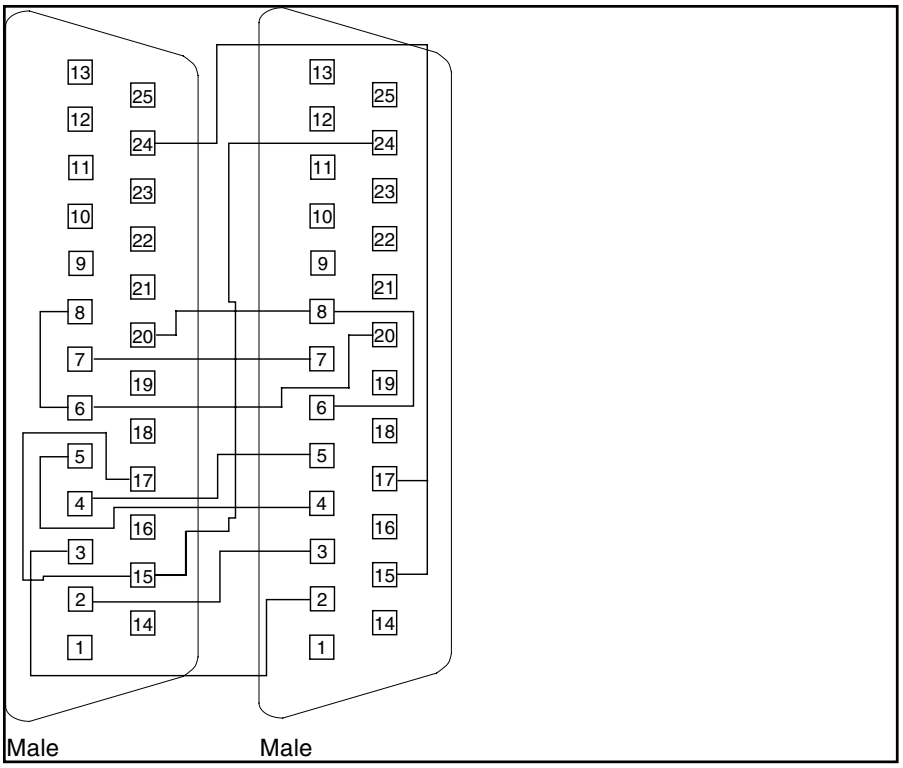


FIGURE A-2 25-pin RS-232 Port-to-Port Loopback Cable Wiring Diagram

8-Pin to 8-Pin Loopback Cable

FIGURE A-3 shows how to connect 8-pin round DIN RS-232 port to RS-423 to 8-pin round-DIN RS-232 and RS-423 port loopback cables. Both plugs are male. The following table lists the pin connections for connecting the first plug to the second plug:

TABLE A-4 8-Pin to 8-Pin Loopback Cable Wiring Plug Connections

First Plug	Second Plug
Pin 3	Pin 5
Pin 5	Pin 3

TABLE A-4 8-Pin to 8-Pin Loopback Cable Wiring Plug Connections *(Continued)*

First Plug	Second Plug
Pin 6	Pin 2
Pin 2	Pin 6
Pin 7	Pin 1

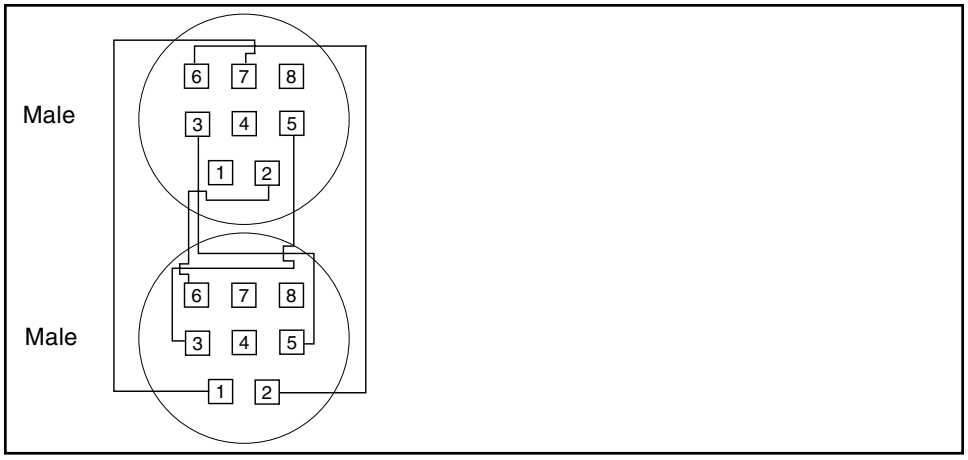


FIGURE A-3 8-Pin to 8-Pin Loopback Cable Wiring Diagram

Pin 8, Receive clock In (DD), remains unconnected.

8-Pin Loopback Plug

FIGURE A-4 shows how to connect male 8-pin round-DIN RS-232 and RS-423 single-port loopback plugs. The following table lists the pin connections for connecting the first plug to the second plug:

TABLE A-5 8-Pin Loopback Plug Wiring Plug Connections

First Plug	Second Plug
Pin 3	Pin 5
Pin 6	Pin 2
Pin 1	Pin 7

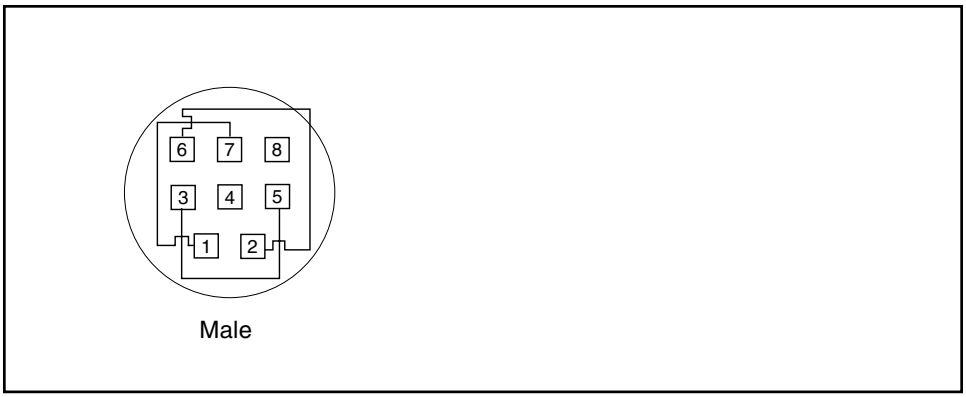


FIGURE A-4 8-Pin Loopback Plug Wiring Diagram

Pin 8, Receive Clock In (DD), remains unconnected.

25-Pin Port A-to-Port B Loopback Plug

FIGURE A-5 shows how to connect a 25-pin Port A to Port B loopback plug for most systems. The following table lists the pin connections for connecting the first plug to the second plug:

TABLE A-6 Port A-to-Port B Loopback Plug Wiring Plug Connections

First Plug	Second Plug
Pin 16	Pin 2
Pin 3	Pin 14
Pin 13	Pin 4
Pin 5	Pin 19
Pins 6 and 8	Pin 11
Pin 12	Pin 20
Pin 18	Pin 24
Pins 15 and 17	Pin 25

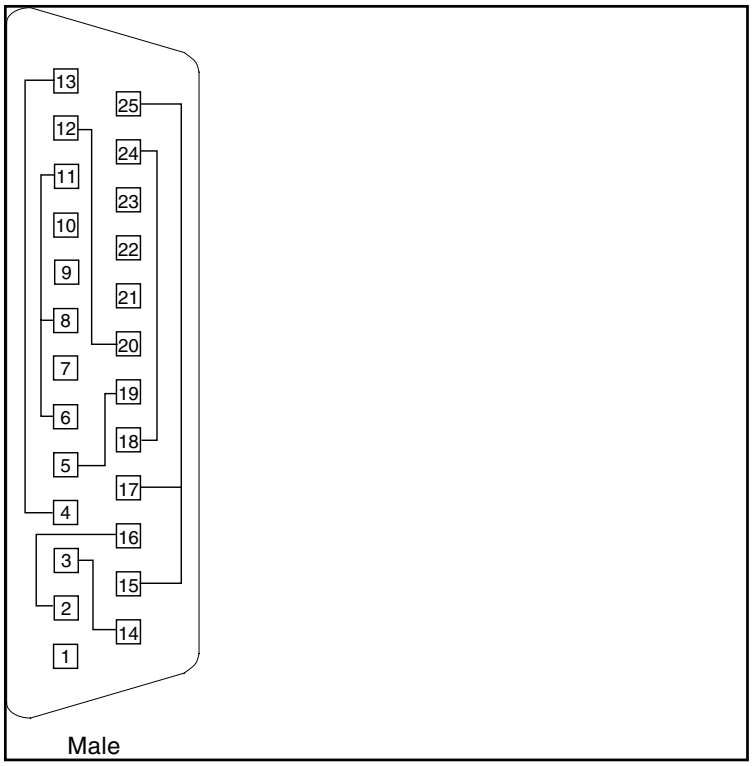


FIGURE A-5 Port A-to-Port B Loopback Plug Wiring Diagram

25-Pin Port A-to-A Port B-to-B Loopback Plug

If your system has a single communications port to connect it to peripherals, follow the connection instructions in FIGURE A-6 to make a male 25-pin loopback plug for that communication port. The following table lists the pin connections for connecting the first plug to the second plug:

TABLE A-7 Port A-to-A, Port B-to-B Loopback Plug Wiring Plug Connections

First Plug	Second Plug
Pin 3	Pin 2
Pin 5	Pin 4
Pins 6 and 8	Pin 20
Pin 12	Pin 11
Pin 13	Pin 19
Pin 16	Pin 14
Pins 15 and 17	Pin 24
Pin 25	Pin 18

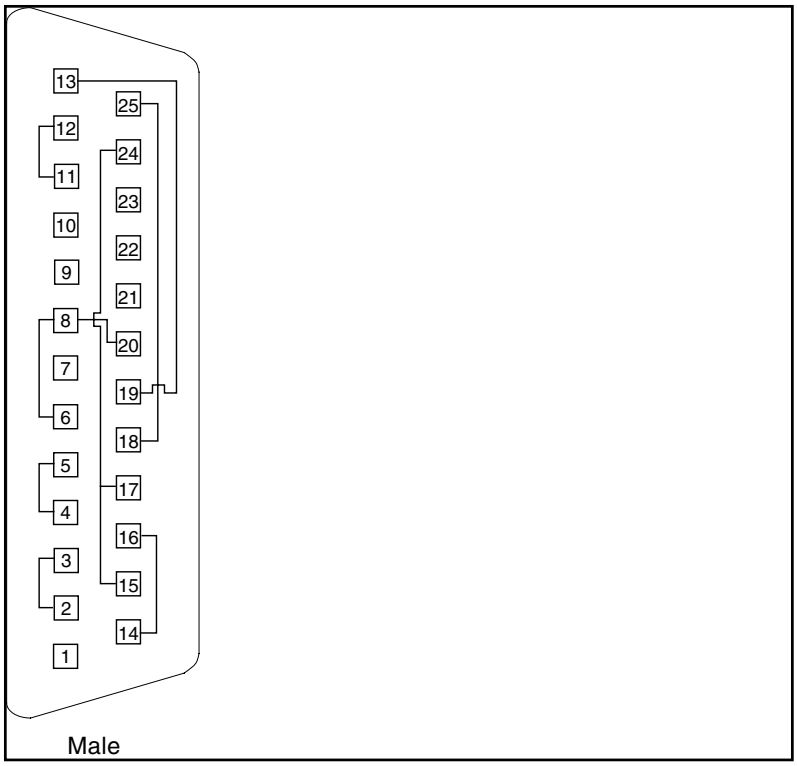


FIGURE A-6 Port A-to-A, Port B-to-B Loopback Plug Wiring Diagram

96-Pin Female Loopback Connector

FIGURE A-7 shows a 96-pin connector that can be ordered from Sun (part number 370-1366). The following table lists the pin connections for connecting the first plug to the second plug:

TABLE A-8 96-Pin Female Loopback Connector Wiring Plug Connections

First Plug	Second Plug
Pins 4 and 12	Pin 77
Pin 5	Pin 20
Pins 6	Pin 36
Pin 10	Pin 11

TABLE A-8 96-Pin Female Loopback Connector Wiring Plug Connections *(Continued)*

First Plug	Second Plug
Pin 13	Pin 16
Pin 14	Pin 15
Pin 18	Pin 19
Pin 21	Pin 24
Pin 28	Pin 60
Pin 29	Pin 68
Pin 30	Pin 34
Pin 37	Pin 40
Pin 38	Pin 39
Pin 42	Pin 43
Pin 45	Pin 48
Pin 46	Pin 47
Pin 52	Pin 78
Pin 53	Pin 55
Pin 54	Pin 75
Pin 58	Pin 59
Pin 61	Pin 64
Pin 62	Pin 63
Pin 66	Pin 67
Pin 69	Pin 72
Pin 76	Pin 79
Pin 82	Pin 83
Pin 85	Pin 88
Pin 86	Pin 87
Pin 90	Pin 91
Pin 93	Pin 96
Pin 94	Pin 95

The following are the materials used for this plug:

- PCR-E96FA(1)
- PCS-E96LKPA(1)
- 3751 Metal Plug(1) (9563K42)

- AWG28 Madison Cable(8″ long) UL/CSA Approved

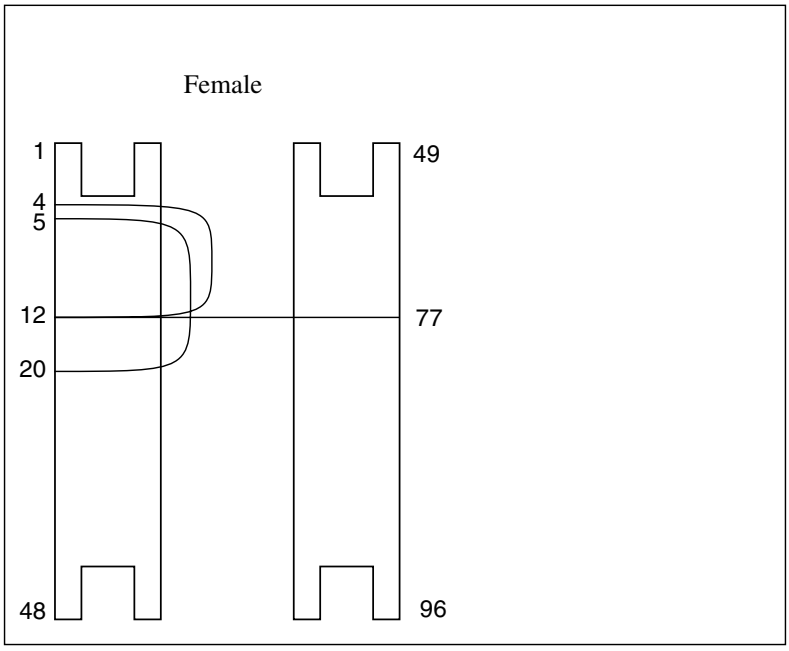
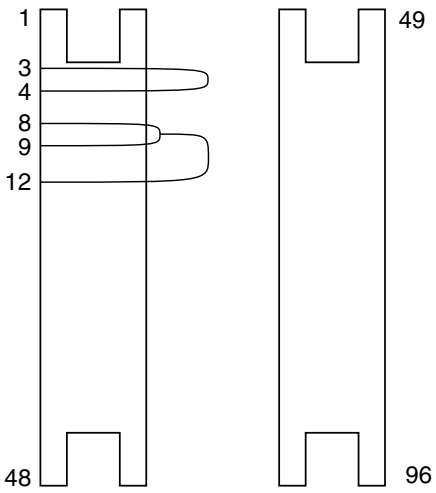


FIGURE A-7 96-Pin Female Loopback Connector Wiring Diagram

96-Pin Female Special Loopback Connector

This 96-pin connector can be ordered from Sun (part number 370-1381).



Materials:
PCR-E96FA(1)
PCS-E96LKPA(1)
9563K999 Cap(1)
Madison Cable(6" long)
(#28 SCSI UL/CSA Approved)

Open Contacts:
1, 2, 6,16, 23, 24, 25, 26, 30, 40, 47, 48, 49, 50, 54, 64, 71, 72, 73, 74, 78, 88, 95, 96

Connect: First Connector	to	Second Connector
pin 3		pin 4
pin 5		pin 7
pins 8 and 9		pin 12
pin 10		pin 11
pin13		pin 14
pin 15		pin 17
pins 18 and 19		pin 22
pin 20		pin21
pin 27		pin 28
pin 29		pin 31
pins 32 and 33		pin 36
pin 34		pin 35
pin 37		pin 38
pin 39		pin 41
pins 42 and 43		pin 46
pin 44		pin 45
pin 51		pin 52
pin 53		pin 55
pins 56 and 57		pin 60
pin 58		pin 59
pin 61		pin 62
pin 63		pin 65
pins 66 and 67		pin 70
pin 68		pin 69
pin 75		pin 76
pin 77		pin 79
pins 80 and 81		pin 84
pin 82		pin 83
pin 85		pin 86
pin 87		pin 89
pins 90 and 91		pin 94
pin 92		pin 93

FIGURE A-8 96-Pin Female Special Loopback Connector Wiring Diagram

37-Pin RS-449 Loopback Cable

Use these wiring instructions for a loopback cable for two 37-pin RS-449 synchronous ports.

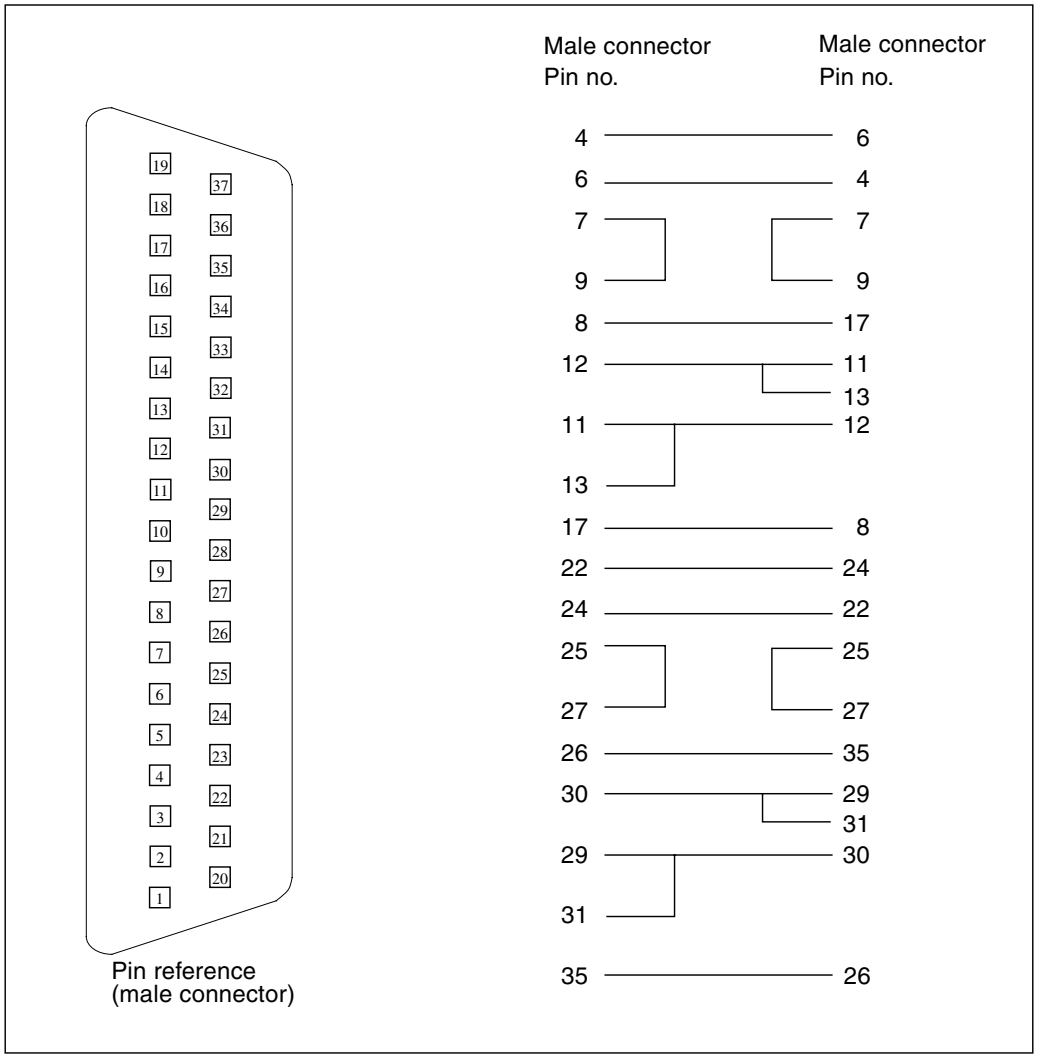


FIGURE A-9 37-Pin RS-449 Loopback Cable Wiring Diagram

37-Pin RS-449 Loopback Plug

Use these wiring instructions to make a male 37-pin RS-449 loopback plug. This plug is also available from Sun (part number 530-1430).

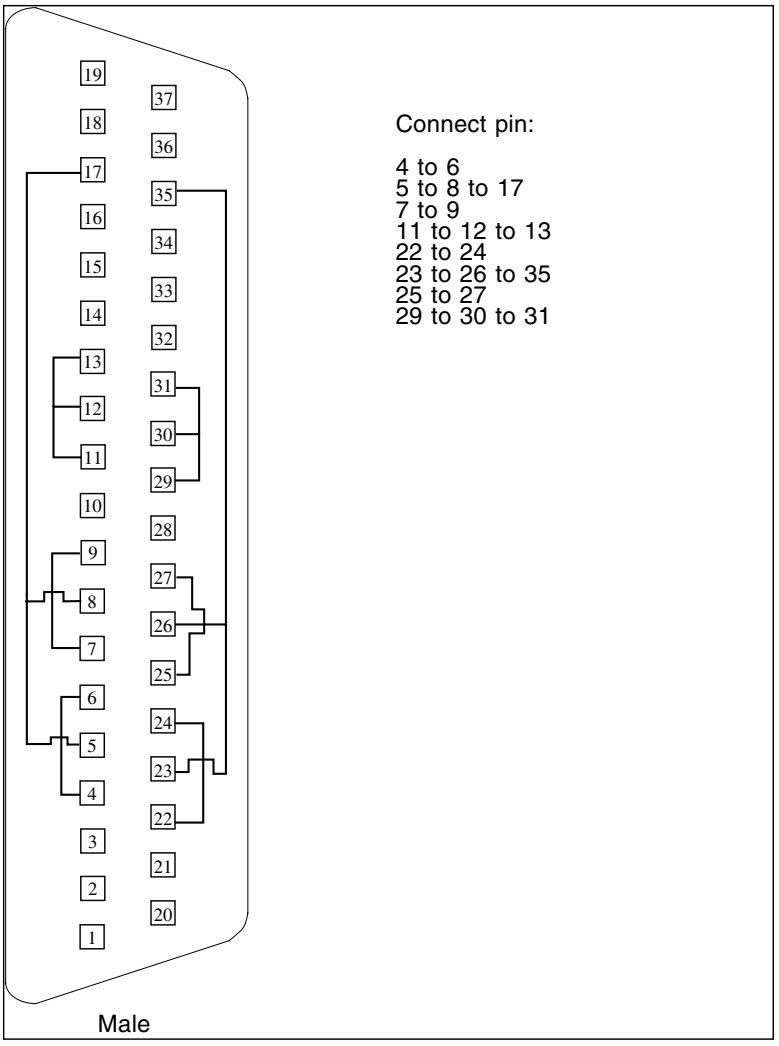


FIGURE A-10 37-Pin RS-449 Loopback Plug Wiring Diagram

9-Pin Male Single-Port Loopback Plug

Use these wiring instructions for male 9-pin RS-232 and RS-423 single-port loopback plugs.

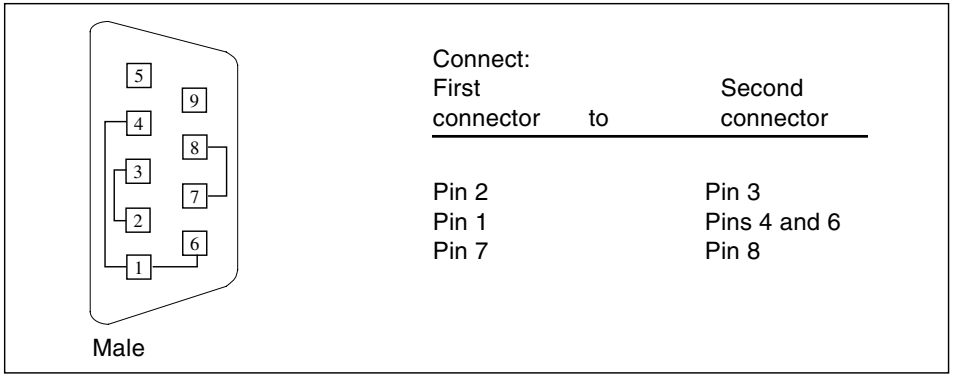


FIGURE A-11 9-Pin Male Single-Port Loopback Plug Wiring Diagram

9-Pin Female Single-Port Loopback Plug

Use these wiring directions for female 9-pin RS-232 and RS-423 single-port loopback plugs. Use this loopback plug with the `pcmciatest`.

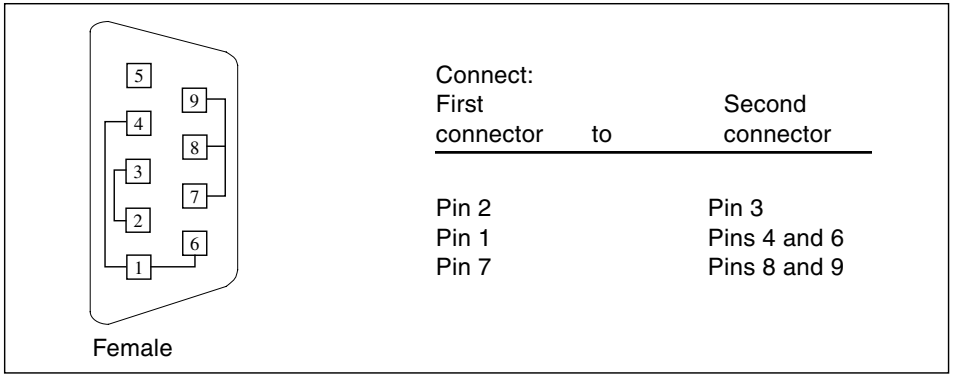
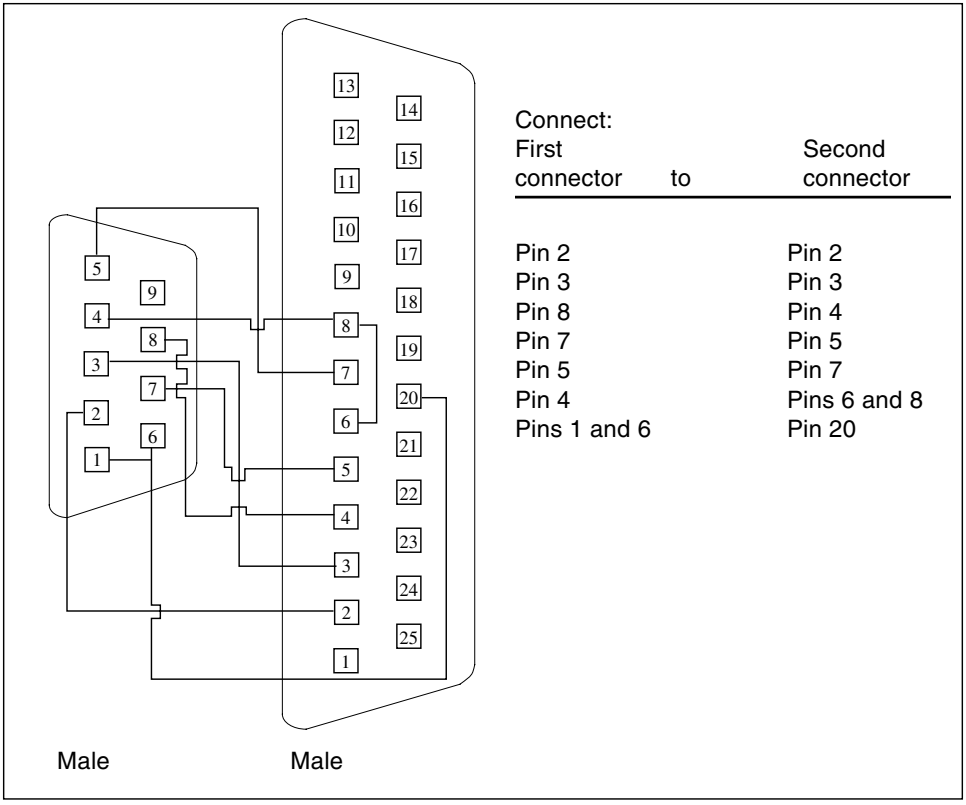


FIGURE A-12 9-Pin Female Single-Port Loopback Plug Wiring Diagram

9-Pin to 25-Pin Port-to-Port Loopback Cable

Use these wiring instructions for a 9-pin RS-232 and RS-423 port to 25-pin RS-232 and RS 423 port loopback cables. Both connectors are male.



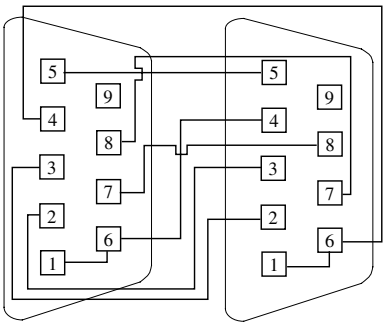
Connect:

First connector	to	Second connector
Pin 2		Pin 2
Pin 3		Pin 3
Pin 8		Pin 4
Pin 7		Pin 5
Pin 5		Pin 7
Pin 4		Pins 6 and 8
Pins 1 and 6		Pin 20

FIGURE A-13 9-Pin to 25-Pin Port-to-Port Loopback Cable Wiring Diagram

9-Pin to 9-Pin Port-to-Port Loopback Cable

Use these wiring instructions for 9-pin RS-232 and RS 423 port to 9-pin RS-232 and RS-423 port loopback cables. Both plugs are male.



Connect: First connector	to	Second connector
Pins 1 and 6		Pin 4
Pin 2		Pin 3
Pin 3		Pin 2
Pin 4		Pins 1 and 6
Pin 5		Pin 5
Pin 7		Pin 8
Pin 8		Pin 7

FIGURE A-14 9-Pin to 9-Pin Port-to-Port Loopback Cable Wiring Diagram

This cable has no Sun part number assigned to it.

NT to TE Loopback Cable

Using two standard RJ-45 connectors, and connect pin 1 to pin 1, pin 2 to pin 2, and so on, for all pins. This loopback is a straight-through connection.

Twisted-Pair Ethernet (TPE) Loopback Cable for Fast Ethernet

Use the following wiring instructions for standard RJ-45 connectors for Fast Ethernet. Use loopback cable in `netlbtest` for `eri` devices.

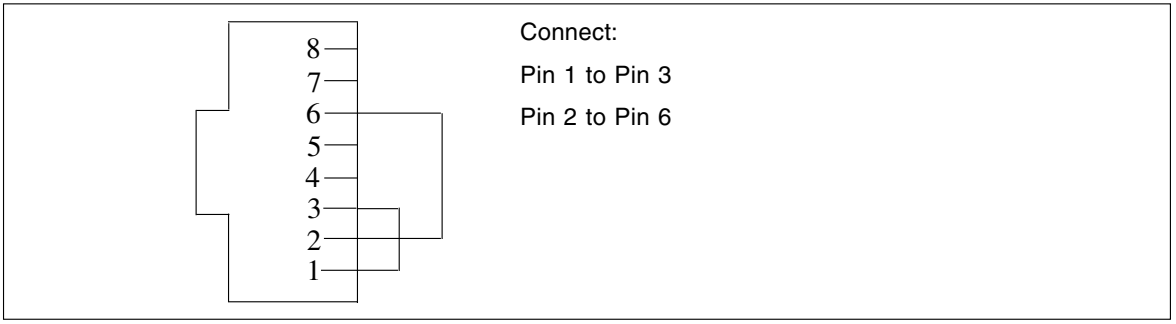


FIGURE A-15 Twisted-Pair Ethernet (TPE) Loopback Cable Wiring Diagram for Fast Ethernet

TPE Loopback Cable for Gigabit and 10/100 Ethernet

Use the following wiring instructions for RJ-45 plugs for Gigabit and 10/100 Ethernet. This loopback cable is used in `netlbtest` for Gigabit and 10/100 devices.

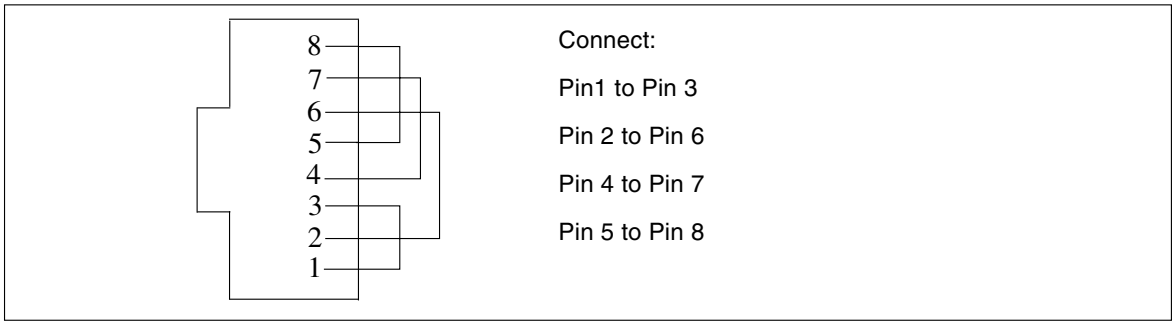


FIGURE A-16 TPE Loopback Cable Wiring Diagram for Gigabit and 10/100 Ethernet

x86 Platform RJ-45 Serial Port Loopback Connector

Use the following wiring instructions for x86 platform RJ-45 serial port loopback plugs.

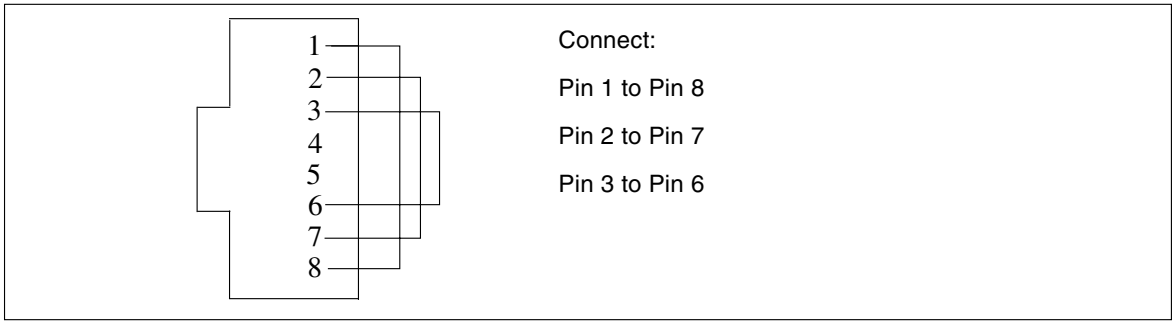


FIGURE A-17 x86 Platform RJ-45 Serial Port Loopback Connector

9-Pin Male Single-Port Loopback Plug

Use these wiring instructions for male 9-pin RS-232 and RS-423 single-port loopback plugs.

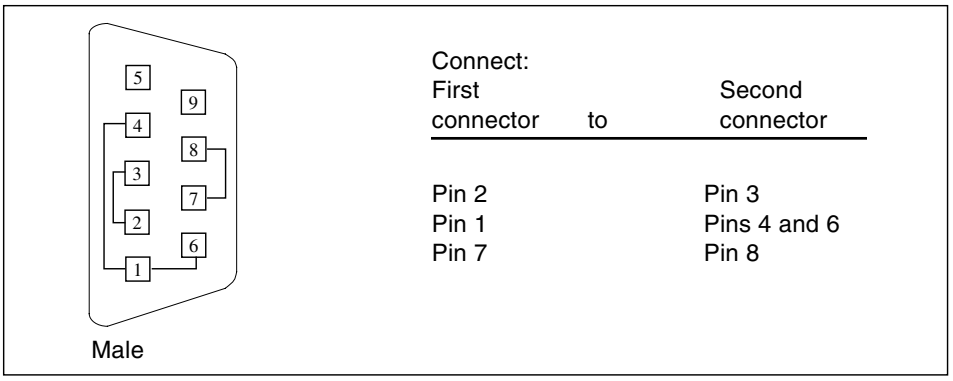


FIGURE A-18 9-Pin Male Single-Port Loopback Plug Wiring Diagram

9-Pin Female Single-Port Loopback Plug

Use these wiring directions for female 9-pin RS-232 and RS-423 single-port loopback plugs. Use this loopback plug with the `pcmciatest`.

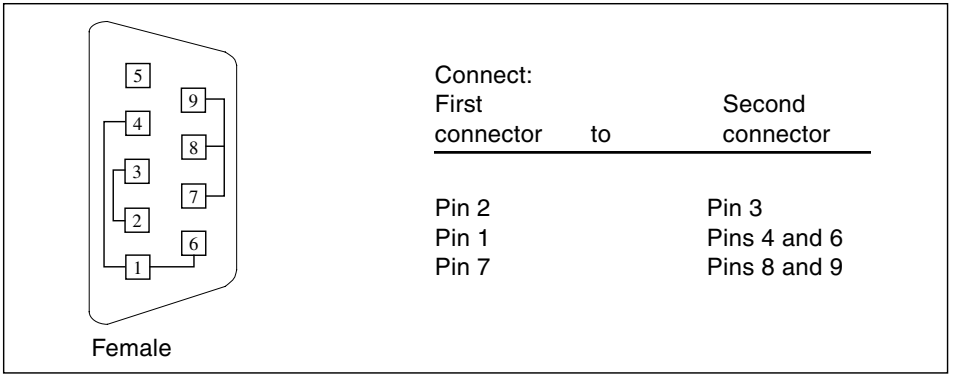


FIGURE A-19 9-Pin Female Single-Port Loopback Plug Wiring Diagram

9-Pin Male DB-9 External Loopback Connector

Use these wiring instructions for male 9-pin DB-9 external loopback plugs.

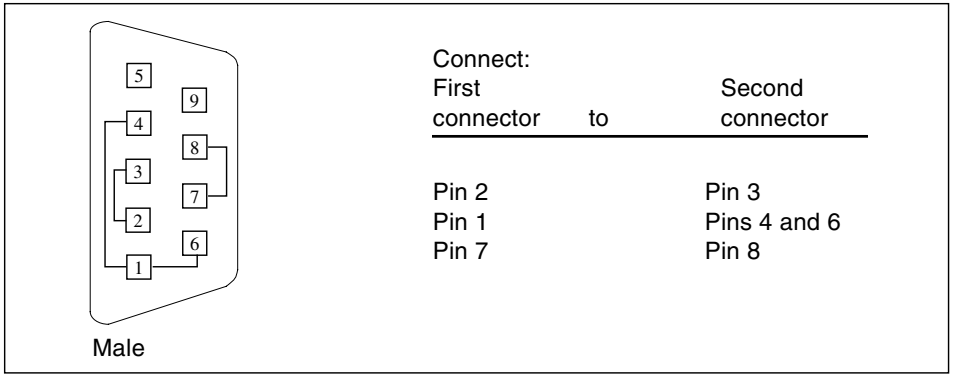


FIGURE A-20 9-Pin Male DB-9 External Loopback Connector Wiring Diagram

The signals and names for the DB-9 connector are as follows.

Pin	Name	Signal
1	DCD	Data Carrier Detect
2	RxD	Receive Data
3	TxD	Transmit Data
4	DTR	Data Terminal Ready
5	SGND	Signal Ground
6	DSR	Data Set Ready
7	RTS	Request To Send
8	CTS	Clear To Send
9	RI	Ring Indicator

9-Pin Female DB-9 External Loopback Connector

Use these wiring instructions for female 9-pin DB-9 external loopback connectors. Use this loopback connector with the `netlbtest` on x86 platforms.

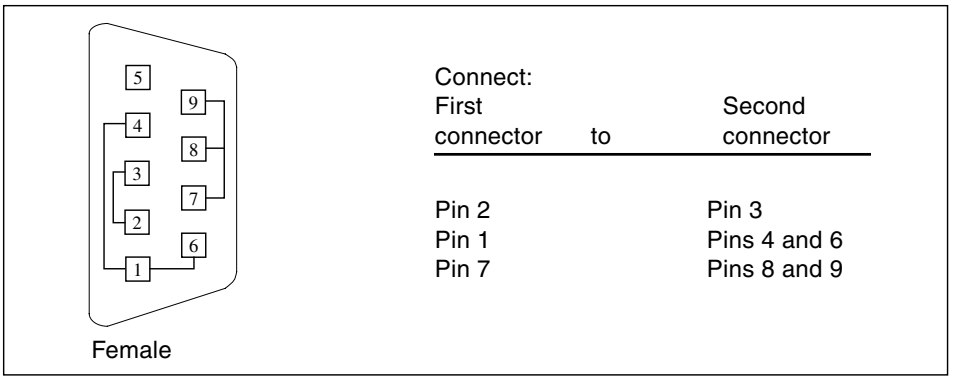


FIGURE A-21 9-Pin Female DB-9 External Loopback Connector Wiring Diagram

Glossary

administrative domain An arbitrary collection of hosts and networks that are monitored by the software as a single hierarchal entity. You may choose to divide their enterprise into several domains, each to be managed by different users.

agent A software process, usually corresponding to a particular local managed host, that carries out manager requests and makes local system and application information available to remote users.

bus A point-to-point network component. Used by the software to represent a network link to which many other hosts may be connected.

community A string similar to a password that is used to authenticate access to an agent's monitored data.

COMA Cache Only Memory Architecture. In a COMA machine, additional hardware including tag and state memory is added to the DRAM of each processing node to convert it into a kind of cache called attraction memory. See also NUMA.

CTQ Critical to Quality. A Sun Sigma terminology, used to identify key issues for a product based on voice of the customer.

diagnosis Correct interpretation and reporting of error.

diagnosability Ability of system to detect and correctly report errors when they occur.

diagnostics A test to uncover faults. These tests are used in design and validation to find margins and set limits on error protection, and stress tests used in operations to screen finished product for escapes.

diagnostic harness The harness or environment that schedules, manages, and controls execution of diagnostic tests on a platform.

DRAM Dynamic Read Access Memory. A type of semiconductor random access memory that stores information in integrated circuits that contain capacitors. Because capacitors lose their charge over time, the dynamic RAM must be periodically "refreshed" or recharged. Contrast with SRAM. See also VRAM.

EEPROM Electronically Erasable Programmable Read Only Memory. A special type of PROM that can be erased by exposing it to an electrical charge. Like other types of PROM, EEPROM retains its contents even when the power is turned off. Also like other types or ROM, EEPROM is not as fast as RAM. See also PROM, ROM, and RAM.

FC-AL Fibre Channel-Arbitrated Loop A connector that provides high bandwidth, increased distance, and additional connectivity from host to peripherals.

FIFO First-In First-Out Memory that stores data in queue order so the first input element goes out the first.

FRU Field Replaceable Unit.

GUI The graphical user interface, or GUI, is a window that provides the user with a method of interacting with the computer and its special applications, usually with a mouse or other selection device. The window usually includes such things as windows, an intuitive method of manipulating directories and files, and icons.

HDLC High Level Data Link. A common layer two protocol that handles both data and control messages.

hop The number of routers a packet goes through before reaching its destination.

module A software component that may be loaded dynamically to monitor data resources of systems, applications and network devices.

node A node is a workstation or server.

OpenBoot Diagnostics Standalone operation, without native operating system. Interactive menu driven, which provides ability to run tests from OpenBoot PROM level.

offline diagnostics Diagnostics that require the aid of the native operating system to function. Primarily tests components not currently in use by customer.

online diagnostics Diagnostics that require the aid of the native operating system to function. Primarily tests components currently in use by customer.

NUMA Non-Uniform Memory Access. CC/NUMA or CC-NUMA is Cache-Coherent Non-Uniform Memory Access. In a CC-NUMA machine, the physical address on the memory bus of a processing node is used to determine the home node memory location of a particular datum. The cache hierarchy on each processing node is constructed to replicate and hold copies of data from not only the local memory, but also the memory of remote nodes.

PICL Platform Information and Control Library.

POST Power on Self Test. Standalone operation, without native operating system. Mostly non interactive, automated tests, run when power is applied to the system.

production environment One of two environments in which software is deployed. The production environment is a "real" environment (as opposed to a test environment) in which you manage and monitor your hardware.

PROM Programmable Read Only Memory A memory chip on which data can be written only once. When a program has been written onto a PROM, it remains there forever. Unlike RAM, PROMs retain their data even when the power is turned off. See also RAM.

RAM Random Access Memory A type of computer memory that can be accessed randomly; that is, any byte of memory can be accessed without touching the preceding bytes. RAM is the most common type of computer and device memory.

ROM Read Only Memory Computer memory on which data has been prerecorded. Once data has been written onto a ROM chip, it cannot be removed and can only be read.

RTS/CTS Ready to Send/Clear to Send A hardware flow control handshake protocol used with serial lines.

SEEPROM Serial EEPROM. See also EEPROM.

segment An object representing a "segment" of the network, and used as a basis for a local network.

SNMP Simple Network Management Protocol. A simple protocol designed to allow networked entities (hosts, routers, and so on) to exchange monitoring information.

SNMPv2 usec SNMP version 2, user-based security model security standards.

SRAM Static Random Access Memory. A faster and more reliable RAM than dynamic random access memory (DRAM). DRAM offers access times of about 60 nanoseconds, while SRAM access can be as low as 10 nanoseconds. It is static because it does not require refreshment as does DRAM. See also VRAM.

standalone diagnostics Diagnostics that run without the aid of the native operating system. Usually an operating environment is built to provide basic scheduling capabilities.

standard error An open file normally connected directly to a primary output device, such as a terminal, printer, or screen. Error messages and other diagnostic output normally go to this file and then to the output device. You can redirect the standard error output into another file instead of to the printer or screen.

standard input (Standard input device) The device from which a program or system normally takes its input. Usually a terminal or the keyboard.

standard output (Standard output device) The device to which a program or system normally sends its output. Usually a terminal or the screen.

URL Uniform Resource Locator. A URL is a textual specification describing a resource which is network-accessible.

VRAM Video Random Access Memory. A type of dynamic RAM (DRAM) used in high-speed graphics frame buffers. With conventional DRAM, both the processor and the frame buffer logic must access RAM by sharing the same signal lines or buses on the RAM chips. VRAM provides separate buses for the processor and the frame buffer logic. See also DRAM, SRAM.

Index

T

V

W