SunVTS™ 4.4 Test Reference Manual

Sun Microsystems, Inc.
901 San Antonio Road
Palo Alto, CA 94303-4900
U.S.A. 650-960-1300

Part No. 816-0796-10
July 2001, Revision A

Send comments about this document to: docfeedback@sun.com

Contents

Figures

Tables

Preface

SunVTS™ 4.4 is the Sun Microsystems™ Validation Test Suite. SunVTS is a comprehensive software diagnostic package that tests and validates Sun™ hardware by verifying the configuration and functionality of most hardware controllers, devices, and platforms.

SunVTS is primarily used from a graphical user interface (GUI), which may be either Common Desktop Environment (CDE) or OPEN LOOK. This book describes SunVTS tests that run on machines with SPARC™ 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 4.4 User's Guide* for overall SunVTS information. Developers or experienced users who want to run the SunVTS diagnostic application will find these documents useful.

Before You Read This Book

In order to make full use of the information in this document, you may need access to the following documents:

- *SunVTS 4.4 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 does not contain information on basic UNIX® commands and procedures such as shutting down the system, booting the system, and configuring devices.

See one or more of the following for this information:

- *Solaris Handbook for Sun Peripherals*
- AnswerBook2™ online documentation for the Solaris™ 8 software environment
- Other software documentation that you received with your system

Typographic Conventions

TABLE P-1 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. Command-line variable; replace with a real name or value.	Read Chapter 6 in the *User's Guide*. These are called *class* options. You *must* be `root` to do this. To delete a file, type `rm` *filename*.

Shell Prompts

TABLE P-2 Shell Prompts

Shell	Prompt
C shell	*machine_name*%
C shell superuser	*machine_name*#
Bourne shell and Korn shell	$
Bourne shell and Korn shell superuser	#

Related Documentation

This manual covers SunVTS revision 4.4 tests. It serves as a reference companion to the SunVTS 4.4 manuals listed below.

TABLE P-3 Related Documentation

Application	Title	Part Number
Installation and Navigation	*SunVTS 4.4 User's Guide*	816-0795-10
Quick Reference	*SunVTS Quick Reference*	816-0861-10

Ordering Sun Documentation

Fatbrain.com, an Internet professional bookstore, stocks select product documentation from Sun Microsystems, Inc.

For a list of documents and how to order them, visit the Sun Documentation Center on Fatbrain.com at:

`http://www.fatbrain.com/documentation/sun`

Accessing Sun Documentation Online

The `docs.sun.com`[SM] web site enables you to access Sun technical documentation on the Web. You can browse the `docs.sun.com` archive or search for a specific book title or subject at:

`http://docs.sun.com`

Sun Welcomes Your Comments

We are interested in improving our documentation and welcome your comments and suggestions. You can email your comments to us at:

`docfeedback@sun.com`

Please include the part number (816-0796-10) of your document in the subject line of your email.

CHAPTER **1**

Introduction

This manual describes SunVTS™ Version 4.4 tests that are distributed on the Sun Computer Systems Supplement CD.

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

SunVTS is composed 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.

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

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

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 and requirements. For overall test configuration modes and options refer to the *SunVTS 4.4 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 4.4 User's Guide* for installation information.

Test Requirements

SunVTS version 4.4 was first introduced and designed to run in the Solaris 8 7/01 operating environment. It is recommended that you run SunVTS 4.4 in the Solaris 8 7/01 operating environment.

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 environments. 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/sparcv9/`*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 4.4 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 8, the following command can be used 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 and OPEN LOOK (OL) 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 6). TABLE 1-1 describes the various SunVTS user interfaces. Refer to the *SunVTS 4.4 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 or OPEN LOOK (see note below) 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	Lets users 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 – The SunVTS OPEN LOOK user interface does not support the latest SunVTS features and will be discontinued when the OPEN LOOK environment is discontinued in the Solaris operating environment. For full feature support, use the SunVTS CDE or TTY interface. Refer to the Solaris "End-of-Software Support Statements" section of the Solaris operating environment release notes for the latest end of support news.

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. Press 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, OPEN LOOK, 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.

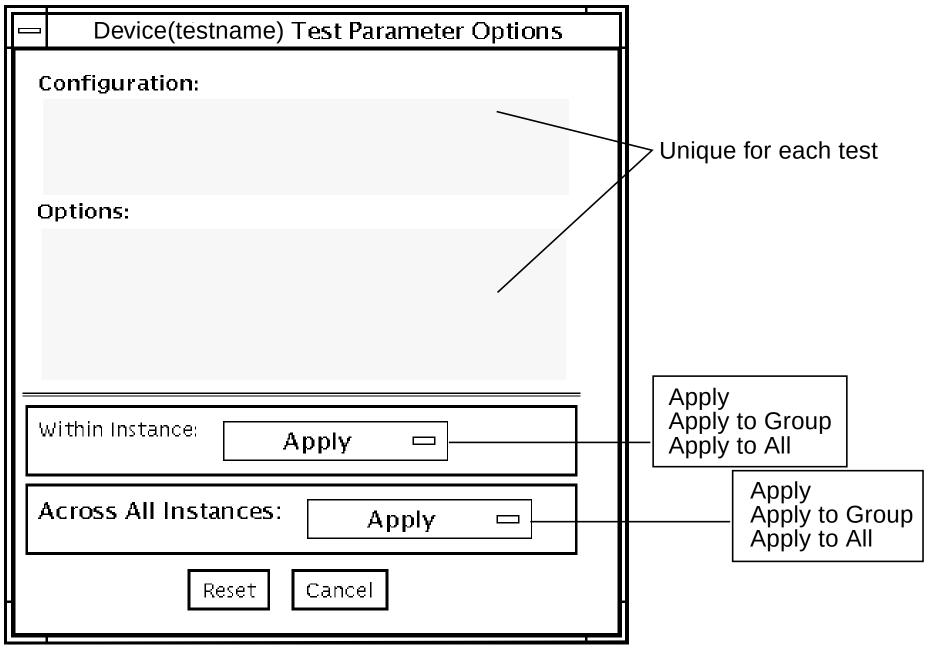


FIGURE 1-1 Test Parameter Options Dialog Box (CDE)

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.

Note – The Test Parameter Options Dialog Box descriptions also apply to the OPEN LOOK interface. However, the Apply to Group and Apply to All buttons are not available in the SunVTS OPEN LOOK user interface.

Note – The SunVTS OPEN LOOK user interface may not support the latest SunVTS features. For full feature support, use the SunVTS CDE or TTY interface. The SunVTS OPEN LOOK user interface will be sustained, but not enhanced, as long as OPEN LOOK is supported in the Solaris environment.

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*. 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 “32-Bit and 64-Bit Tests” on page 3.

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.
`-u`	Displays command-line usage information.
`-v`	Runs the test in Verbose mode and displays messages with more detailed information about the testing process. The default is false.
`-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 test Trace mode and displays messages that track function calls and sequences currently in use by the test code. The default is false.
`-e`	Runs the test in Stress mode by increasing the system load. 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.

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

Argument	Description
`-n`	Runs the test in Connection mode. See the note below. The default is false.
`-f`	Runs the test in full Functional test mode. This mode assumes that the test has complete control of the device under test. See the note below. The default is false.
`-i` *number*	Defines the number of instances for scalable tests.
`-w` *number*	For scalable tests, defines to which instance the test is assigned.
`-o`	Used to indicate 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 each 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 or OPEN LOOK 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:

```
# ./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 you move the mouse, you will receive false error messages. Even a slight mouse movement can cause a test to fail.

Both the screen saver option and the system save/resume option of Power Management must be disabled on the system being tested before conducting any frame buffer test.

If you are using either the CDE or the Open Windows 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 or OPEN LOOK). 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 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

Advanced Frame Buffer Test (afbtest)

afbtest verifies the functionality of the advanced frame buffer.

afbtest can detect and adapt to the various video modes of the advanced frame buffer (AFB). Instead of only running in one standard graphics mode, all tests can run in any mode. In stereo mode, all tests write into the right and left eyes unless you specify otherwise.

You can interrupt afbtest using Control-c. Turn off all other keyboard input if OPEN LOOK is running on the unit being tested.

Test accuracy is checked using a checksum algorithm. Possible locations of failing pixels are colored chartreuse to help visually identify their position.

Caution – Do not run any other application or screen saver program that uses the AFB accelerator port while running afbtest. This combination causes SunVTS to return incorrect errors.

afbtest Test Requirements

Disable all screen savers before testing any graphics device. Type **xset s off** at a UNIX® prompt to disable the Solaris screen saver.

For full instructions on testing frame buffers, please see "Testing Frame Buffers" on page 9.

afbtest requires approximately 29 Mbytes of disk space in the /tmp directory to extract its working files. If this space is not available, the diagnostic will fail and report warning and error messages, indicating a lack of disk space.

Note – Do not run Open Windows across multiple monitor while running `afbtest`, otherwise the test will return errors.

`afbtest` Options

To reach the 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 may not include the device appropriate to this test. Refer to the *SunVTS User's Guide* for more details.

By default, all `afbtest` options are enabled.

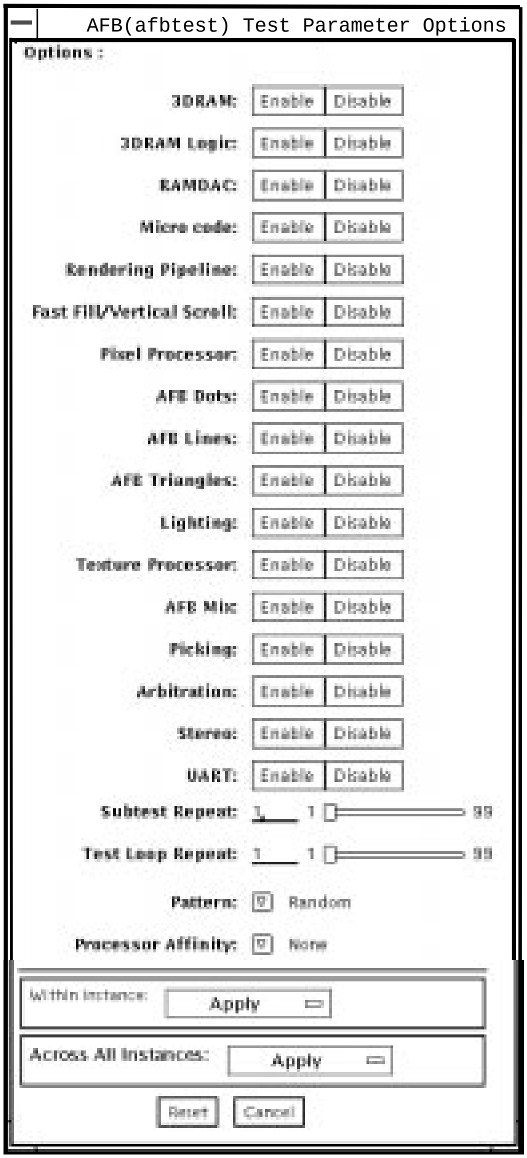


FIGURE 2-1 `afbtest` Test Parameter Options Dialog Box

TABLE 2-1 `afbtest` Options

`afbtest` **Options**	**Description**
3DRAM test	The 3DRAM test thoroughly tests the video memory in the AFB using 512-bit reads and writes. 3DRAM makes a full-screen pass, consisting of a write and a read to each pixel location, for each access mode on the list below. The data used can be either random or specified by the user. A second pass is made with the one's complement of the data used in the first pass so that each memory location is tested with both a zero and a one. Errors in this subtest are attributes to the 3DRAM. A failing chip is indicated by (X, Y) locations and device-specific "U" numbers. • DFB8R, DFB8G, DFB8B, DFB8X—Buffer A • DFB24—Buffer A • DFB32—Buffer A • SFB8R, SFB8G, SFB8B, SFB8X—Buffer A • SFB8R, SFB8G, SFB8B, SFB8X—Buffer B • SFB32—Buffer A • SFB32—Buffer B • SFB32—Buffer C • SFB64—Buffers A and C • SFB64—Buffers B and C

TABLE 2-1 `afbtest` Options

`afbtest` **Options**	**Description**
3DRAM Logic test	The 3DRAM Logic test provides logical functionality to the AFB. The following services are tested: • Compare Controls—Match AB • Compare Controls—Magnitude AB • Compare Controls—Match C • Compare Controls—Magnitude C • Match Mask—AB • Magnitude Mask—AB • Match Mask—C • Magnitude Mask—C • Raster Operations—RGB • Raster Operations—X • Raster Operations—YZ • Plane Mask—RGB • Plane Mask—X • Plane Mask—Y • Plane Mask—Z • Group Enable—R, G, B, X • Group Enable—Y, Z Each function is tested separately with a series of SFB64 writes. A total of 16 writes are made for each different test case with Y coordinate values varying from 0 to 30 in increments of 2 pixels. This dotted column organization provides page thrashing and block flashing in all screen resolutions. For each operation all possible combinations are tested. For example, in `ROP RGB` *new==old* there are three possible values: *new < old, new == old, and new > old*. Each of these cases are tested. Five passes of the functions are made. Each pass writes into a different AFB address space: SFB32-A, SFB32-B, SFB32-C, SFB64-AC, and SFB64-BC. Note that the passes that write into the SFB32 address spaces are writing two pixels at a time because the tests use SFB64 writes. Care is taken to ensure that all 3DRAM chips are tested. Errors in this subtest are attributed to the 3DRAM.

TABLE 2-1 `afbtest` Options

`afbtest` **Options**	**Description**
RAMDAC test	RAMDAC registers are tested using simple read / write patterns to determine if there are any bad bits. This includes all LUTs (4 CLUTs, PWLUT and OWLUT). `afbtest` ensures that data is actually being read from the RAMDAC and not being supplied by the driver. RAMDAC on AFB can be in SEP8 or Combined mode. RAMDAC test detects the RAMDAC mode and tests the RAMDAC output for that mode. The RAMDAC Signature Register captures the pixels going to the screen. This test determines that all of the different data paths within the RAMDAC are functioning properly. The data pattern is designed so all the data paths are tested. i.e., All CLUTs, PWLUTs, and OWLUTS. A cursor is also displayed on the screen. Errors in this test are attributed to the RAMDAC.
Microcode test	Microcode test generates the checksum for the microcode of the each enabled float and compares all the checksums for equality. Errors in this test are attributed to the microcode PROMS & SRAMS.
Rendering Pipeline test	Rendering Pipeline test uses the rendering pipeline tests developed for the FFB stand-alone diagnostics. Each FFB primitive is tested thoroughly with a variety of sources and configurations: • Dots • Anti-aliased dots • Lines using all four line drawing primitives • Triangles • Polygons • Rectangles • Fonts Errors in this test are attributed to the Draw Chips.
Fast Fill / Vertical Scroll test	Fast Fill / Vertical Scroll primitives are separated from the Rendering Pipeline tests because of their dependence on screen type. There are three different tests, one for each screen type. Each test uses both block and page mode fast fills. Errors in this test are attributed to the Draw Chips.

TABLE 2-1 `afbtest` Options

`afbtest` **Options**	**Description**
Pixel Process test	The Pixel Processor test, a subtest, exercises the options selected by the AFB's Pixel Processor Control (PPC) register: • Auxiliary clipping (additive and subtractive) • Depth cueing • Alpha blend • Viewport clip (2D and 3D) • Area pattern (transparent and opaque) Errors in this test are attributed to the Draw Chips.
AFB Dots test	This test uses the AFB primitive tests developed for the AFB stand-alone diagnostics. AFB Dots are tested thoroughly with a variety of sources and configurations: • Dots • Anti-aliased dots • Big dots Errors in this test are attributed to the Command & Draw Chips.
AFB Lines test	This test uses the AFB primitive tests developed for the AFB stand alone diagnostics. AFB Lines are tested thoroughly with a variety of sources and configurations: • Jaggy lines • Anti-aliased lines • Lines with patterns • Bresenham lines • Wide lines drawn as lines and triangles Errors in this test are attributed to the Command & Draw Chips.
AFB Triangles test	This test uses the AFB primitive tests developed for the AFB stand alone diagnostics. AFB Triangles are tested thoroughly with a variety of sources and configurations: • Triangles drawn clock wise & counter clockwise • Triangles drawn as stripes • Independent triangles • Triangles drawn as stars • Triangles with facet normals Errors in this test are attributed to the Command & Draw Chips.

TABLE 2-1 `afbtest` Options

`afbtest` **Options**	**Description**
Lighting test	The Lighting test exercises AFB float and lighting microcode. This test lights an object with maximum number of lights (32) that AFB can handle in hardware. A checksum is generated for the rendered image and compared with the checksum generated for the same image on a known good system. Errors in this test are attributed to the Float & Microcode SRAMS.
Texture Processor test	The Texture Processor test exercises the different options of the AFB's Texture Pixel Processor Control (TPPC) register: • Texture Minification • Texture Magnification • Blend • Decal • Modulation Errors in this test are attributed to the Draw Chips.
AFB Mix test	The AFB Mix test draws different primitives with variety combinations of sources and configurations, exercising all the Draw, Float, Microcode, and 3DRAM chips on AFB. This test is to stress the AFB. Errors in this test are attributed to Draw, Float, Microcode, and/or 3DRAM Chips.
Picking test	The Picking test exercises the pick detect login of the 3DRAM. We define a pick detect window and make sure that writes to the window are picked, and writes outside the window are not picked. The test is repeated once for each 3DRAM. Errors in this test are attributed to the 3DRAM.
Arbitration test	The Arbitration test, a subtest, continuously renders an object into the accelerator port while doing reads and writes through the direct port. A picture is rendered into all 32 planes of the B buffer while the other process does 32-bit DFB reads and writes in the A plane. This subtest simulates conditions in the real world, where rendering processes and windows operations run concurrently. Errors in this test are attributed to the Context switching between DFB and SFB.

TABLE 2-1 `afbtest` Options

`afbtest` **Options**	**Description**
Stereo test	Stereo test displays an object in stereo mode with different images for the right and left eye. The user can verify proper operation by looking at the screen with stereo glasses and following the instructions being displayed. If the monitor type is not 1280x1024 at 76MHz, this test prints a warning message and does not execute. To prevent this message from being displayed or written to the SunVTS information log, disable the stereo test in the Test Parameter Options dialog box. Only Sony P4 and N2 monitors support stereo resolutions. This test temporarily switches the monitor into stereo mode, renders a stereo image, performs a signature analysis on the stereo image (using the RAMDAC signature capture register), and after displaying the image for five seconds, restores the monitor to its previous resolution. Errors in this test are attributed to the RAMDAC.
UART test	The UART test tests both UART0 and UART1. First, UART memory is tested using simple read/write patterns to determine if there are any bad bits. Then data is written to UART 0/1 and the written data is read using the internal loopback in polling mode. The read data is verified with written data. Errors in this test are attributed to UART and its SRAM memory chip.

`afbtest` Test Modes

Due to the nature of graphic tests, reading data from, or writing data to the frame buffer during graphic tests will disturb user operation. For this reason, `afbtest` is only available in offline Functional test mode.

TABLE 2-2 `afbtest` Test Modes

Test Mode	**Supported?**	**Description**
Connection	No	Not supported.
Functional (Offline)	Yes	Runs the full set of tests.

`afbtest` Command-Line Syntax

`/opt/SUNWvts/bin/afbtest` *standard_arguments* **`-o dev=`***device_name*`,`
`S=`*subtest_number*`,`**`F=`***#_of_subtest_loops*`,`**`B=`***#_of_test_loops*`,`**`P=`***test_pattern*

TABLE 2-3 `afbtest` Command-Line Syntax

Argument	Description
dev=*device_name*	*device_name* is the relative path name of the device being tested with respect to `/dev/fbs`; The default is `afb0`.
S=*subtest_number*	*subtest_number* is the test number of the subtest to be run. Select from the subtests below. You can run multiple subtests by adding the subtest numbers together. For example, *n=0x3* runs both test 1 and test 2; *n=0x180* runs both test *0x080* and test *0x0100*. You do not need the leading zeros. • *n*—`0x00001 3DRAM` • *n*—`0x00002 3DRAM Logic` • *n*—`0x00004 RAMDAC` • *n*—`0x00008 Micro code` • *n*—`0x00010 Rendering Pipeline` • *n*—`0x00020 FastFill/Vertical Scroll` • *n*—`0x00040 Pixel Processor` • *n*—`0x00080 AFB Dots` • *n*—`0x00100 AFB Lines` • *n*—`0x00200 AFB Triangles` • *n*—`0x00400 Lighting` • *n*—`0x00800 Texture Processor` • *n*—`0x02000 AFB Mix Test` • *n*—`0x04000 Picking` • *n*—`0x08000 Arbitration` • *n*—`0x10000 Stereo` • *n*—`0x40000 UART`
F=*#_of_subtest_loops*	The number of times to repeat each subtest. The default is 1.
B=*#_of_test_loops*	The number of times to repeat a test loop before passing. The default is 1.
P=*test_pattern*	The test pattern number. The default is **`r`**, for random patterns. You may also choose `0` for `0x0000000`, `3` for `0x3333333, 5` for `0x5555555`, or `9` for `0x9999999`.

Note – 64-bit tests are located in the `sparcv9` subdirectory: `/opt/SUNWvts/bin/sparcv9/`*testname*. If the test is not present in this directory, then it may only be available as a 32-bit test. For more information refer to “32-Bit and 64-Bit Tests” on page 3.

Note – Errors returned by `afbtest` are nonspecific: It is not possible to determine which component caused a failure. In all error conditions, the field replaceable unit (FRU) is the entire AFB.

CHAPTER 3

Alarm Card Test (alarmtest)

The alarmtest exercises the Alarm Card on the Sun Netra™ ct 400 and ct 800 systems.

The Alarm Card is a hot-swappable add-on option for the Netra ct systems which provides secure remote access for system monitoring, failure recovery, and alarm notification. The alarm card can be used in both front- and rear-access systems.

This test is not scalable.

alarmtest disables the envmond daemon. After running alarmtest, restart this daemon using the command:

```
% /etc/init.d/envmon start
```

Note – Do not run alarmtest and rsctest at the same time. Tests may return incorrect results.

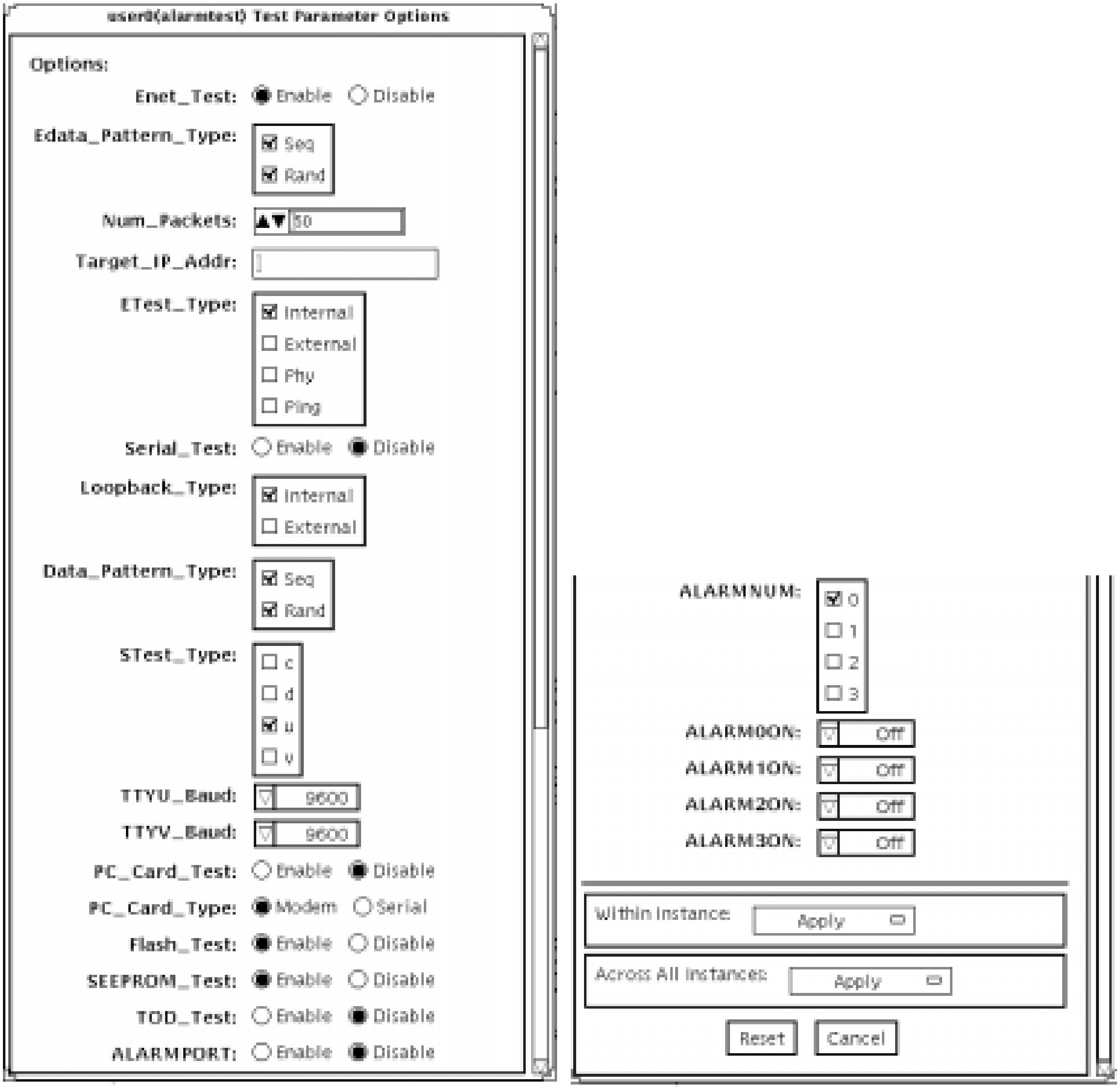
`alarmtest` Subtests

`alarmtest` consists of seven subtests which test and report on the following:

- Ethernet
- Serial ports
- PC-card (PCMCIA) socket
- Flash memory
- SEEPROM
- TOD
- Alarmport

`alarmtest` Options

To reach the 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 may not include the device appropriate to this test. Refer to the *SunVTS User's Guide* for more details.



Test Parameter view, scrolled up

Test Parameter view, scrolled down

FIGURE 3-1 `alarmtest` Test Parameter Options Dialog Box

TABLE 3-1 `alarmtest` Options

Option	Description
Enet_Test	Enables or disables Ethernet testing.
Edata_Pattern_Type	Selects the type of data pattern for `Enet_Test`: Sequential, Random, or both.
Num_Packets	Specifies the number of data packets to be sent in one test loop.
Target_IP_Addr	Specifies the IP address of a host to use for the ping test.
Etest_Type	Selects any or all internal, external, Phy (ethernet transceiver), or ping tests.
Serial_Test	Enables or disables `serial_test`.
Loopback_Type	Selects internal loopback, external loopback, or both.
Sdata_Pattern_Type	Selects the type of data pattern for `serial_test`: Sequential, Random, or both.
STest_Type	Selects ports to be tested: `c`, `d`, `u`, or `v`.
TTYU_Baud	Selects the alarm card's COM1 port baud rate.
TTYV_Baud	Selects the alarm card's COM2 port baud rate.
PC_Card_Test	Enables or disables PC card (PCMCIA) testing.
PC_Card_Type	Specifies the card type for the PC Card: modem or serial i/o.
Flash_Test	Enables or disables the flash checksum test.
SEEPROM_test	Enables or disables the SEEPROM checksum test.
TOD_test	Enables or disables the TOD checksum test.
ALARMPORT	Enables or disables the alarmport test.
ALARMNUM	Selects any or all alarm ports to be tested: 0, 1, 2, 3.
ALARM0ON	Turns on, turns off, or toggles (on then off) alarm port 0.
ALARM1ON	Turns on, turns off, or toggles (on then off) alarm port 1.
ALARM2ON	Turns on, turns off, or toggles (on then off) alarm port 2.
ALARM3ON	Turns on, turns off, or toggles (on then off) alarm port 3.

Note – The alarmport test requires a visual check using an LED box.

`alarmtest` Loopbacks

The loopback tests use the following external loopbacks:

- Ethernet loopback test—standard RJ-45 connector. Connect pin 1 to pin 3, and pin 2 to pin 6.
- Serial loopback test for Netra ct 800—DB-9 connector. Connect pin 2 to pin 3, pins 4 and 6 to pin 1, and pin 7 to pin 8.
- Serial loopback test for Netra ct 400—RJ-45. Connect pin 6 to pin 3, pin 1 to pin 8, and pin 2 to pin 7.

`alarmtest` Test Modes

TABLE 3-2 `alarmtest` Test Modes

Test Mode	Supported?	Description
Connection	Yes	Reports the status of the alarm card.
Functional (Offline)	Yes	Runs the subtests for Ethernet, Serial, Flash, PCMCIA, SEEPROM, TOD, and Alarmport.

`alarmtest` Command-Line Syntax

`/opt/SUNWvts/bin/alarmtest` *standard_arguments* **`-o enet=`***E(nable)/D(isable)*,**`epattype=`***seq+rand*,**`epkts=`***number_packets*,**`target=`***IP_address*,**`etest=`***I+E+H+P*,**`serial=`***E(nable)/D(isable)*,**`slb=`***I+E*,**`spattype=`***seq+rand*,**`stest=`***c+d+u+v*, **`pccard=`***E(nable)/D(isable)*,**`pccardtype=`***modem/serial*,**`flash=`***E(nable)/D(isable)*,**`seeprom=`***E(nable)/D(isable)*,**`tod=`***E(nable)/D(isable)*,

ttyubaud=*ALL | specific_baud***,ttyvbaud=***ALL | specific_baud***,**
aport=*[E]nable | [D]isable***,anum=***0+1+2+3***,a0on=***On | Off/ | T]oggle***,**
a1on=*On | Off | [T]oggle***,a2on=***On | Off | [T]oggle***,a3on=***On | Off | [T]oggle*

TABLE 3-3 `alarmtest` Command-Line Syntax

Argument	Explanation
enet=*E(nable)/D(isable)*	Enables or disables Ethernet testing.
epattype=*seq+rand*	Selects the type of data pattern for `Enet_Test`: Sequential, Random, or both.
epkts=*number_packets*	Specifies the number of data packets to be sent in one test loop.
target=*IP_address*	Specifies the IP address of a host to use for the ping test.
etest=*I+E+H+P*	Selects any or all internal, external, Phy, or ping tests.
serial=*E(nable)/D(isable)*	Enables or disables `serial_test`.
slb=*I+E*	Selects internal loopback, external loopback, or both.
spattype=*seq+rand*	Selects the type of data pattern for `serial_test`: Sequential, Random, or both.
stest=*c+d+u+v*	Selects ports to be tested: `c`, `d`, `u`, or `v`.
pccard=*E(nable)/D(isable)*	Enables or disables PC card (PCMCIA) testing.
pccardtype=*modem/ serial*	Specifies the card type for the PC Card: modem or serial i/o. Default is modem.
flash=*E(nable)/D(isable)*	Enables or disables the flash checksum test.
seeprom=*E(nable)/ D(isable)*	Enables or disables the SEEPROM checksum test.
tod=*E(nable)/D(isable)*	Enables or disables the TOD checksum test.
ttyubaud=*ALL | specific_baud*	Defines baud rates to be used in testing the alarmcard's COM1 port.
ttyvbaud=*ALL | specific_baud*	Defines baud rates to be used in testing the alarmcard's COM2 port.
aport=*[E]nable | [D]isable*	Enables or disables the alarmport test.
anum=*0+1+2+3*	Selects any or all alarm port to be tested: 0, 1, 2, 3
a0on=*On | Off/ | T]oggle*	Turns on, turns off, or toggles (on then off) alarm port 0.
a1on=*On | Off | [T]oggle*	Turns on, turns off, or toggles (on then off) alarm port 1.
a2on=*On | Off | [T]oggle*	Turns on, turns off, or toggles (on then off) alarm port 2.
a3on=*On | Off | [T]oggle*	Turns on, turns off, or toggles (on then off) alarm port 3.

Note – 64-bit tests are located in the `sparcv9` subdirectory: `/opt/SUNWvts/bin/sparcv9/`*testname*. 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 “32-Bit and 64-Bit Tests” on page 3.

CHAPTER 4

SunATM Adapter Test (atmtest)

atmtest checks the functionality of the SunATM™-155 and SunATM-622 SBus and PCI bus adapters.

It runs only in loopback (external or internal) mode. The asynchronous transfer mode (ATM) adapter, and ATM device driver must be present. To run the atmtest in external loopback mode, a loopback connector must be attached to the ATM adapter. The internal loopback mode does not require a loopback connector.

atmtest uses DLPI RAW mode to talk to the device driver. It establishes a virtual circuit (VC) to send a message, receive a message, and compare messages. If the message does not match, or the message is out of sequence, it displays an error message.

Using a random number generator, atmtest sends data into a data buffer and then sends each message from a different starting point. This assures that no two consecutive messages are the same.

atmtest can test more than one virtual circuit. The more virtual circuits used increases the stress level of the test. atmtest automatically selects the virtual circuit number which is unique to the test.

atmtest is nonscalable as it provides for more than one virtual circuit to be tested for each instance, which is adequate for the purpose of a stress test.

atmtest Test Requirements

atmtest can only be selected when the Intervention mode is enabled since it requires a loopback connector for external loopback testing. While Intervention mode is enabled, atmtest and nettest are both available as default selections, however, you must deselect nettest when testing the ATM device.

Bring the ATM interface down to make sure that the interface is in offline mode before running `atmtest`.

Note – Do not run `nettest` while running `atmtest`.

`atmtest` Options

To reach the 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 may not include the device appropriate to this test. Refer to the *SunVTS User's Guide* for more details.

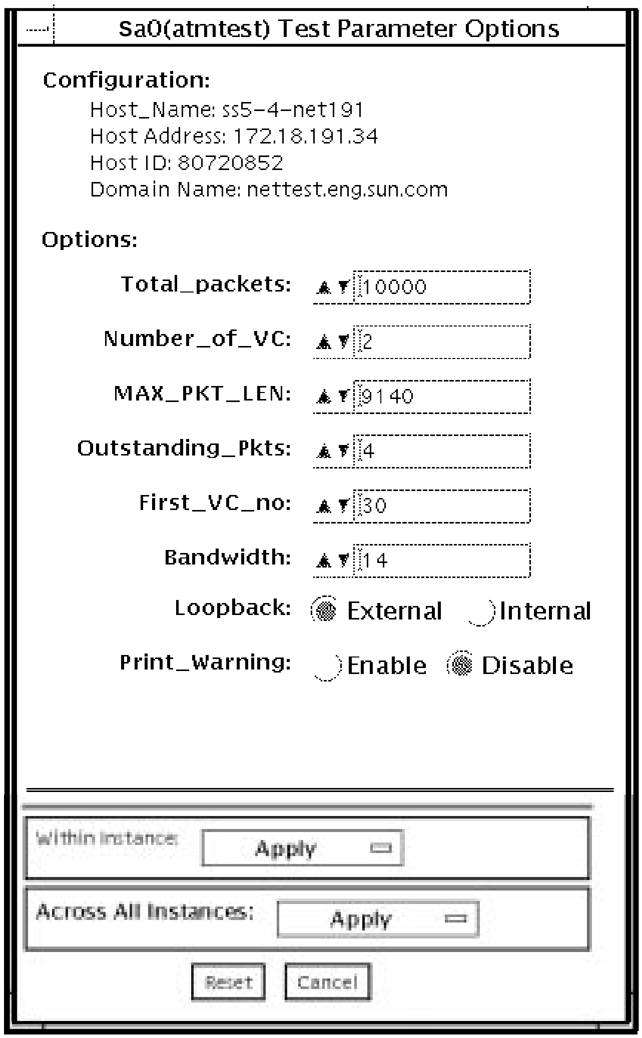


FIGURE 4-1 `atmtest` Test Parameter Options Dialog Box

TABLE 4-1 `atmtest` Options

Options	Description
Configuration	The post address, host ID, and domain name of the system being tested.
Total packets	The total number of packets sent. The default number of packets sent is 10000.
Number of VC	The number of virtual circuits to be set up by one instance. The default number of virtual circuits is 2 for each instance. The `atmtest` uses these two virtual circuits to send out messages simultaneously. The message is received in sending order.
Loopback	Enables the user to select either the external loopback field or internal loopback field. The default selection is the external loopback field. A loopback connector is only needed for external loopback testing.
MAX_PKG_LEN	The maximum packet length to be used by the test to send out the data. The default number is 9140.
Outstanding_pkts	Describes the maximum number of outstanding packets. `atmtest` stops sending messages when the outstanding packet count is more than the number of packets this field specifies.
First_VC_no	Enables the user to set up the starting virtual circuit number to be used for each `atmtest` instance. `atmtest` can automatically avoid virtual circuit numbers that have already been used.
Bandwidth	Enables the user to select different bandwidths to test. The default number is 14.

`atmtest` Test Modes

TABLE 4-2 `atmtest` Test Modes

Test Mode	Supported?	Description
Connection	No	Not supported.
Functional (Offline)	Yes	Runs the full set of tests.

`atmtest` Command-Line Syntax

`/opt/SUNWvts/bin/atmtest` *standard_arguments*,**`-o`**
`dev=`*device*,**`tpkts=`***n*,**`nv=`***n*,**`ml=`***n*,**`bw=`***n*,**`opkts=`***n*,*ld*,*sd*,*sl*,*nc*,*ns*,*vcf*

TABLE 4-3 `atmtest` Command-Line Syntax

Argument	Description
`dev=`*device*	Specifies the device name to be tested, such as `ba0` or sa0.
`tpkts=`*n*	Specifies the number of packets to loopback [1.2147483647].
`nv=`*num_vc*	Specifies the number of simultaneous virtual circuits to be tested.
`ml=`*max_len*	Specifies the maximum length of the random packet.
`bw=`*bandwidth*	Specifies the bandwidth in MBits/s of a virtual circuit.
`opkts=`*n*	Specifies the number of packets for each virtual circuit that can be transmitted without receiving a corresponding packet.
ld	The internal loopback mode is selected.
sd	Changes the payload data to static instead of random.
sl	Changes all packets to their maximum length.
nc	Instructs the test not to check the receive payload (improves throughput).
ns	Instructs the test not to exit on a packet reception failure.
vcf=n	Specifies the first virtual circuit number used.

Note – 64-bit tests are located in the `sparcv9` subdirectory: `/opt/SUNWvts/bin/sparcv9/`*testname*. If the test is not present in this directory, then it may only be available as a 32-bit test. For more information refer to “32-Bit and 64-Bit Tests” on page 3.

CHAPTER 5

Audio Test (`audiotest`)

The `audiotest` verifies the hardware and software components of the `audio` subsystem. This test supports all Sun audio implementations.

This test will work with exclusive access devices (only one process or application available at a time), or with newer audio devices which support the software mixer feature available in the Solaris 8 operating environment.

Note – `audiotest` turns the mixer off automatically at run time. Shut down all audio applications before running `audiotest`, as Online mode is not supported. The mixer is restored after testing.

This test is not scalable.

The availability of the following subtests depends on the particular `audio` implementation being tested.

`audiotest` Subtests

TABLE 5-1 `audiotest` Subtests

Subtest	Description
Record/Play test	This test plays and records one second of data. It does not check data. This test is run on all audio implementations.
Crystal test	The crystal test measures the accuracy of the crystal that generates the sample rate clock. It does this by playing a one-second signal and then measuring the actual time required to play the signal. This measurement is performed for each of the eight standard sample rates. This test is available for `dbri`(7) and `audiocs`(7) audio implementations
Loopback tests	This test verifies the functionality and signal quality of the audio ports. The test simultaneously plays and records a known signal. The recorded signal is analyzed for loop gain and signal-to-noise ratio plus distortion. This is repeated at various sample rates, encodings, precisions and channels. The audio ports that are supported depend on the audio implementation under test. The `audiocs(7)` implementation supports loopbacks from/to headphone, line-out, microphone, and line-in ports. The `dbri(7)`/speakerbox implementation supports fewer ports. The `audioamd(7)`implementation does not support loopback tests. Most tests require a stereo loopback cable. Note: The microphone loopback tests require special hardware and are used by manufacturing centers and special test facilities. Do not invoke the microphone loopback tests unless you have the required hardware.
Controls test	This test verifies the three control buttons on the Sun speakerbox. The Controls test plays music while the user is prompted to press the Volume Down, Volume Up, and Mute buttons in a specific order. If no button is pressed in 30 seconds the test fails. This test is only supported on the `dbri`(7)/speakerbox implementation.
Audio test	This test plays a 30-second music file out of the speaker or headphone. The full benefit of this test is only realized if the user listens to the output. Badly distorted audio or inaudible music indicates a problem. This test is supported on all audio implementations.

audiotest Options

To reach the 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 may not include the device appropriate to this test. Refer to the *SunVTS User's Guide* for more details.

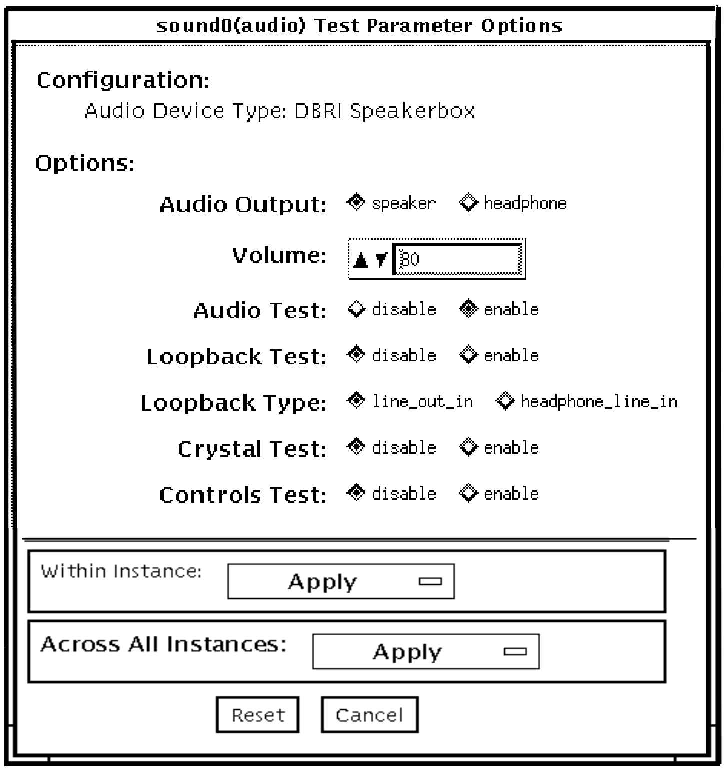


FIGURE 5-1 `audiotest` Test Parameter Options Dialog Box

Note – Upon startup, the SunVTS probe utility determines which audio implementation is present and adjusts the `audiotest` Option menu appropriately. Your dialog box may look different than the one pictured here, but will contain some or all of these options.

Note – The internal loopbacks are only active if the audio jacks are unused (nothing connected).

Some options can only be selected through the command line. See the command-line option descriptions in “audiotest Command-Line Syntax” on page 41.

TABLE 5-2 `audiotest` Options

Option	Description
Audio Output	Selects the output port for the Music Play test.
Volume	Sets the volume for the Music Play test.
Audio test	Enables or disables the Music Play test. This test is enabled by default.
Loopback test	Enables or disables the Loopback test. A loopback cable must be installed between the selected ports to run external loopback tests. This test is disabled by default.
Loopback type	Selects the type of Loopback test to run.
Crystal test	Enables or disables the Crystal test. This test is disabled by default.
Controls test	Enables or disables the speakerbox Controls test. This is an interactive test. The user is prompted to press the control buttons on the speakerbox. This test is disabled by default.

Note – Do not run the Crystal test while running other SunVTS tests. The Crystal test is timing-dependent. If the system is too busy, it fails due to time-out errors.

`audiotest` Test Modes

TABLE 5-3 `audiotest` Test Modes

Test Mode	Supported?	Description
Connection test	Yes	A simple open and close is performed. No data is transferred. The test returns a pass if the device can be opened and closed successfully. If the device cannot be opened because it is busy, then it is assumed that the device is successfully connected to another process and the test passes.
Functional (Offline)	Yes	The record / play test is run and you can choose to run any of the tests described earlier. In this mode, the test will fail if the device is busy.

`audiotest` Command-Line Syntax

`/opt/SUNWvts/bin/audiotest` *standard_arguments* **`-o dev=/dev/sound/`***unit_no***`,I=/dev`***ioctl_device*, **`M,L,Q,S,T=`***loopback_test_type*, **`X,E,LE,CD,CDD=`***CD_device_name***`,CDT=`***track_number***`,CDG=`***play_gain*, **`CDL=`***play_time***`,W,MF=`***filename***`,TF=`***filename*

TABLE 5-4 `audiotest` Command-Line Syntax

Argument	Description
`dev=/dev/`*audio_device*	Specifies the audio device to be tested. The default is `dev=/dev/audio`.
`I=/dev/`*ioctl_device*	Specifies the `audio ioctl` device to be tested. The default is `/dev/audioctl`.
`M`	Enables the Music Play test.
`L`	Enables the Loopback test.
`Q`	Enables the Quality test. This option does the same thing as `L` option except that it prints an extra status message upon completion.
`S`	Enables the speakerbox Controls test.

TABLE 5-4 `audiotest` Command-Line Syntax *(Continued)*

Argument	Description
T=*loopback_test_type*	Specifies the type of Loopback test. The default is 1; the choices are listed below: • 0—Codec Internal Loopback (CS4231 audio only) • 1—Line-in / Line-out • 2—Headphone / Line-in • 3—Headphone / Microphone • 4—Speaker / CD-input • I1—Internal Line-in / Line-out • I2—Internal Spk / Mic • I3—Internal Headphone / Aux1 • I4—Internal Speaker / Aux1 • I5—Internal Headphone / Mic Note: Test type 0 is always run by default on CS4231 audio implementations. Test types 3 and 4 require special hardware, and are used by manufacturing centers and special test facilities. Do not invoke these tests unless you have the required hardware.
X	Enables the Audio Crystal test.
E	Continues testing if an error occurs.
LE	Loops on error. This plays the signal data in a continuous loop.
CD	Enables the `cdtest`. This is for systems with an internal CD-ROM drive. A CD-ROM with music tracks must be loaded prior to running this test.
CDD=*CD_device_name*	Specifies the raw device name for the CD-ROM drive. The default is `CDD=/dev/rdsk/c0t6d0s0`.
CDT=*number*	Specifies the track number of the CD-ROM to play. The default is to play the first track on the disc.
CDG=*play_gain*	Specifies the play gain of the CD Play test (0 to 255). The default is 120.
CDL=*play_time*	Specifies the number of seconds to run the CD Play test. The default is 30 seconds.
W	Shows warning messages during the Loopback test.
MF=*filename*	Selects an optional music file.
TF=*filename*	Specifies an optional tolerance file. Note: The tolerance file is used by manufacturing centers and special test facilities. Do not use this option unless you are familiar with the tolerance file format.

Note – 64-bit tests are located in the `sparcv9` subdirectory: `/opt/SUNWvts/bin/sparcv9/`*testname*. If the test is not present in this directory, then it may only be available as a 32-bit test. For more information refer to “32-Bit and 64-Bit Tests” on page 3.

CHAPTER **6**

Bidirectional Parallel Port Printer Test (bpptest)

bpptest verifies the functionality of the bidirectional parallel port. SBus printer cards have two printer ports: one for any SPARCprinter™ device and one for a parallel port printer.

The bpptest verifies that your SBus card and its parallel port are working properly by attempting to transfer a data pattern from the SBus card to the printer.

Two indications show that the card and printer are functioning properly: First, you can see from the SunVTS Status window that bpptest made a successful pass, and second, that the pattern transmitted to the printer printed correctly.

If the bpptest passes successfully, you know that the SBus DMA circuitry, the printer, and the device driver are functioning properly.

Note – Large PostScript™ files or raster files may require that the printer has 2 Mbytes or more of memory. Otherwise, the printout may appear on two different sheets of paper.

---

bpptest Hardware and Software Requirements

The SBus printer card and device drivers must be installed to run bpptest. A printer must be connected to the SPARCprinter or bidirectional parallel port, and be powered-up. If both a SPARCprinter and a parallel port printer are connected to the SBus card, you can test both devices at the same time.

Note – For a SPARCstation 10, SPARCstation LX, or SPARCclassic™ system, you can connect a printer directly to the onboard parallel port to run `bpptest`.

If you are testing the SPARCprinter port, be sure the magnets on the SPARCprinter paper tray are set to the correct paper size. For more information, see the *SPARCprinter Installation and User's Guide* and the label on the paper tray.

`bpptest` Options

To reach the 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 may not include the device appropriate to this test. Refer to the *SunVTS User's Guide* for more details.

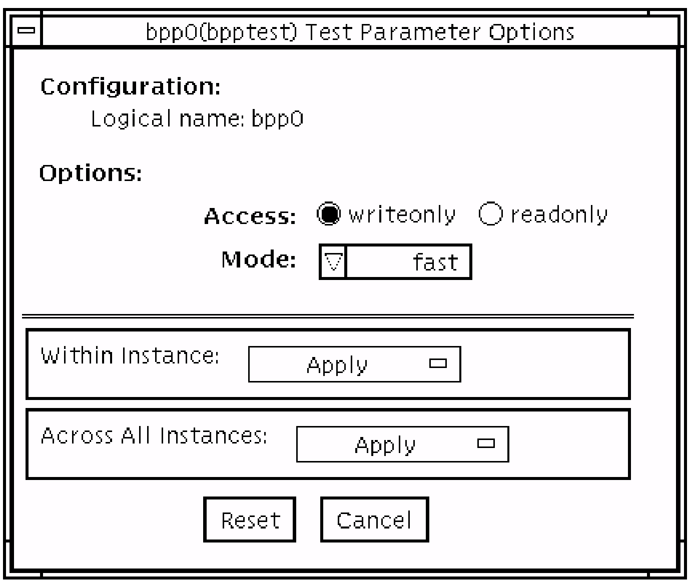


FIGURE 6-1 `bpptest` Test Parameter Options Dialog Box

TABLE 6-1 `bpptest` Options

Option	Description
Access	Determines the direction of data transfer; this field is informational only. `Writeonly` is the only option currently available. Data is transferred only from the SBus printer card to the printer.
Mode	Sets the print intervals. This option lets you select the intervals at which the test image is printed. The default setting is fast. In the online and Functional tests this option is set to medium. In the Functional test, the choices are: • `Fast`—prints an image every 10 seconds. • `Medium`—prints an image every 12 minutes. • `Extended`—prints an image every 30 minutes.

`bpptest` Test Modes

This test supports Connection and Functional test modes.

TABLE 6-2 `bpptest` Test Modes

Test Mode	Supported?	Description
Connection	Yes	In this mode, `bpptest` verifies that a bidirectional parallel port is configured on the system. The success of the `bpptest` in this mode indicates that the bidirectional parallel port hardware and the software driver are installed on the system.
Functional (Offline)	Yes	The testing done in this mode registers a failure if the port is found busy. This is because SunVTS tests make the assumption that all the resources will be available for testing in the Functional test and therefore the unavailability of the device is interpreted as an indication of a fault condition.

`bpptest` Command-Line Syntax

`/opt/SUNWvts/bin/bpptest` *standard_arguments* **`-o`**
`dev=`*device_name***`,access=`***writeonly* | *readonly***`,mode=`***mode*

TABLE 6-3 `bpptest` Command-Line Syntax

Argument	Description
`dev=`*device_name*	Specifies the name of the device. This should be of the form `/dev/bpp`*#*, where # is the minor number of the device.
`access=`*writeonly* | *readonly*	Determines test mode: *writeonly* or *readonly*. Note: This flag is mandatory.
`mode=`*mode*	Sets the test image print rate. The test image is a continuous printout of the ASCII character set. Possible rates are: • `Fast`—prints the test image at 10-second intervals. • `Medium`—prints the test image at 12-minute intervals. • `Extended`—prints the test image at 30-minute intervals.

Note – 64-bit tests are located in the `sparcv9` subdirectory: `/opt/SUNWvts/bin/sparcv9/`*testname*. 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 “32-Bit and 64-Bit Tests” on page 3.

CHAPTER 7

Compact Disc Test (`cdtest`)

`cdtest` checks the CD-ROM unit by reading the CD.

`cdtest` is not a scalable test.

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.

Note – Load a compact disc into the drive before starting the test. See the explanation of CD types in TABLE 7-1.

Volume Management and Compact Discs

`cdtest` tests the CD-ROM drive(s) even if the Volume Manager is not running. If the Volume Manager is running and no media is installed in the CD-ROM 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 a CD in the drive.

cdtest Options

To reach the 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 may not include the device appropriate to this test. Refer to the *SunVTS User's Guide* for more details.

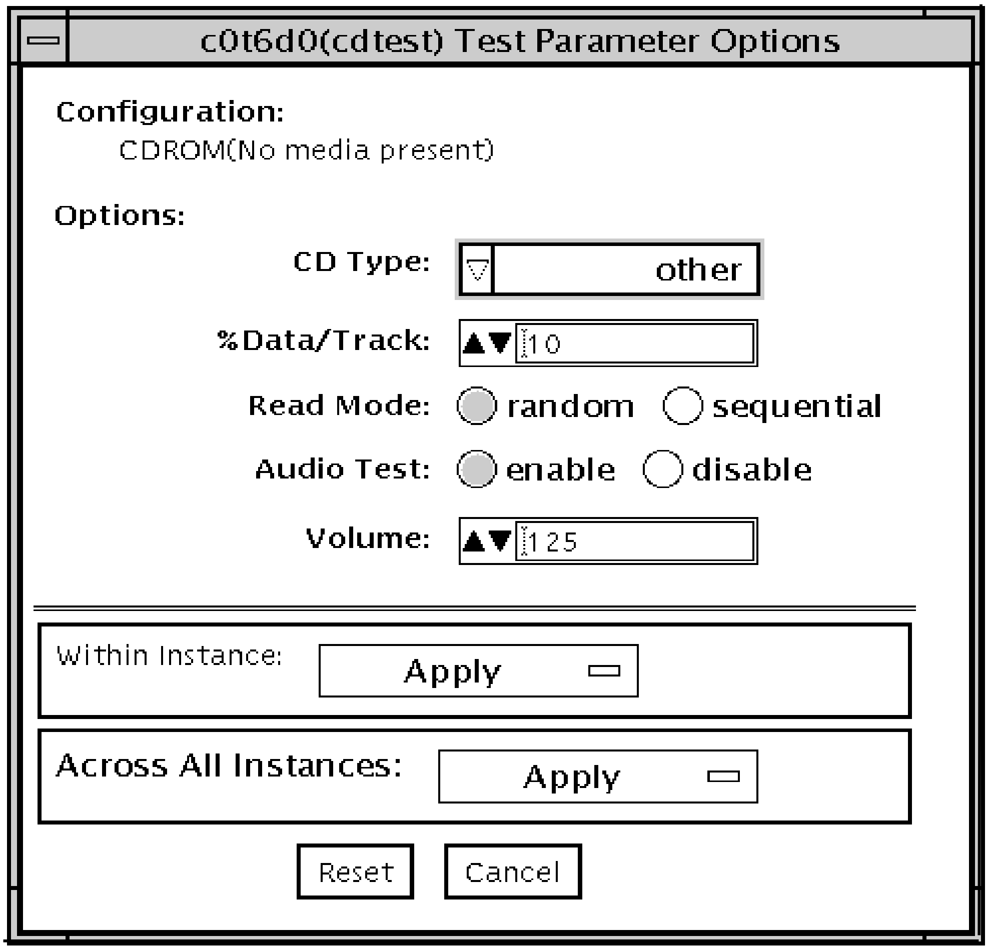


FIGURE 7-1 `cdtest` Test Parameter Options Dialog Box

TABLE 7-1 `cdtest` Options

Options	Description
CD Type	The types of compact discs that can be tested are listed in the CD Type menu. The choices are: `pdo`, `multi-session`, or `other` (the default CD type is `other`).In the Connection test, this option has a default value of other. Note: Your choice must correspond with the disc used 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 canned value of 2%.
Read Mode	`cdtest` 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 canned 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.

`cdtest` Test Modes

This test supports Connection and Functional tests.

TABLE 7-2 `cdtest` Test Modes

Test Mode	Supported?	Description
Connection	Yes	In this mode, `cdtest` verifies that a CD-ROM drive is connected to and configured in the system.
Functional (Offline)	Yes	This mode is similar to Functional online mode except that 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.

`cdtest` Command-Line Syntax

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

TABLE 7-3 `cdtest` 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`**.

Note – 64-bit tests are located in the `sparcv9` subdirectory: `/opt/SUNWvts/bin/sparcv9/`*testname*. 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 "32-Bit and 64-Bit Tests" on page 3.

CHAPTER **8**

Color Graphics Frame Buffer Test (`cg14test`)

`cg14test` checks the `cg14` frame buffer card. `cg14test` is specific to the VSIMM (Video SIMM) devices in the SPARCstation 10 SX and the SPARCstation 20 SX.

Caution – Due to possible conflicts between SunVTS cg14 frame buffer tests and OPEN LOOK applications that use the cg14 frame buffer, the following restrictions apply when running `cg14test`:

Do not run graphic applications other than OPEN LOOK while SunVTS is running frame buffer tests.

Do not run OPEN LOOK programs that generate video updates outside or on top of the SunVTS window.

Do not close the SunVTS window to an icon while it is running frame buffer tests.

`cg14test` Requirements

Ensure that the frame buffer locking option is enabled from the Options window.

For full instructions on testing frame buffers, see “Testing Frame Buffers” on page 9.

cg14test Groups

There are nine test groups in cg14test:

1. MDI and VBC Chip Control Registers
2. Memory Chips
3. MDI Chip Cursor Registers
4. MDI Chip CLUT Registers
5. DAC Chip Registers
6. MDI Chip XLU Registers
7. CG14 Display (visual only)
8. MDI Chip Testmode Readback in 8-bit mode
9. Driver IOCTLs

TABLE 8-1 cg14 Test Groups

Test Groups	Description
Hardware (Groups 1-6)	These test groups are tested by opening /dev/fbs/cgfourteenX, mapping the MDI Control Address Space, modifying the target test location (using direct writes to the mapped address space), reading from the mmapped address space for verification, and closing the device.
Visual Pattern (Group 7)	This subtest loads a visual pattern of 256 colors, then rotates the pattern around. You verify this test by seeing it display.
Data Propagation (Group 8)	This test group is tested by loading the frame buffer (FB) memory with four neutral data patterns, then setting a target FB pixel with data that triggers the test mode readback latch. The result is read from the readback register after vertical blanking occurs. Two different trigger patterns are used at each FB pixel. All four MDI pixel paths (A - D) are used, and the pixel locations for each trigger are designed to detect gross MDI input data opens or short, VRAM SAM addressing, and VRAM-to-SAM transfer addressing. The screen shows four horizontal bars, which are either greyscale or colored. These bars change each time the trigger data is inverted, and as it completes the testing of a raster pattern. Note: If the resolution and VRAM size permits, 8-bits per pixel mode are tested.

TABLE 8-1 `cg14` Test Groups *(Continued)*

Test Groups	Description
Driver (Group 9)	Test all IOCTL calls that have not yet been used to verify proper driver communication to the hardware. Call the driver to perform a hardware update, and then confirm that the update was successful by using the complementary driver read, or reading the mmap'ed address space and comparing it against the stimulus. `cg14test` performs the appropriate steps before and after each test (if possible) to maintain context and prevent visual confusion by saving the register data before it is overwritten, disabling video (if possible), performing the specific test, and restoring the saved register data information. The data used for register testing is optimized to include all 0's, all 1's, and walking a 1 through each bit being tested.
MDI and VBC Chip Control Registers (Group 1)	• Master Control Register bits 7-0 write/read verify • Packed Pixel Register bits 3-0 write/read verify • Master Status Register bits 7-4 read-only verify 0x00 and 0x30 occur • Horizontal Blank Start Register bits 9-0 write/read verify • Horizontal Blank Clear Register bits 9-0 write/read verify • Horizontal Sync Set Register bits 9-0 write/read verify • Horizontal Sync Clear Register bits 9-0 write/read verify • Composite Sync Clear Register bits 9-0 write/read verify • Vertical Blank Start Register bits 11-0 write/read verify • Vertical Blank Clear Register bits 11-0 write/read verify • Vertical Sync Set Register bits 11-0 write/read verify • Vertical Sync Clear Register bits 11-0 write/read verify • Transfer Cycle Set Register bits 9-0 write/read verify (MDI revision 0 only) • Transfer Cycle Clear Register bits 9-0 write/read verify (MDI revision 0 only) • Fault Status Address Register bits 15-0 write/read verify • Auto-increment Address Space Register bits 7-0 write/read verify • Video Base Register bits 23-12 write/read verify
Memory Chips (Group 2)	The Memory Chips test group includes VRAM Testing, Memory Retention, and Test Write Recovery. VRAM Testing: The Data Bus test uses 18 NTA patterns (Nair, Thatte, and Abraham's testing procedure for RAM) to check for data and address faults. This test is performed in `MDI_CHUNKY_XBGR_MAP` access mode only. See TABLE 8-2.

TABLE 8-1 `cg14` Test Groups *(Continued)*

Test Groups	Description
Memory Chips (Group 2) Continued	VRAM Testing (Continued) The test ascends through the frame buffer memory, clearing it to 0's. The NTA pattern test number *x* reads a location to make sure test data *y* is present. It then writes new data *z* to that location. The location ascends through the FB sequentially. See TABLE 8-2. Memory Retention: The VRAM Data Retention test checks for gross problems with the VRAM refresh. Since refresh is active during this test, no retention problems should occur unless the refresh is defective. This test turns off the video, writes 0's to all the VRAM, waits the specified `memory_hold` time (the default is five seconds), then reads and compares all VRAM data. This process is repeated with data of f's, then the video is restored and the test is complete. Two command-line parameters are related to this test: **R=***number* and **H=***number*. **R=** lets the user specify the refresh interval from 128-1023. The time between refresh cycles and the system default is 123. **H=** lets the user specify the retention test hold time in seconds. Write Recovery: A Write Recovery test is used in all the EMC mapping modes to write data to 0's followed by immediately reading that data location to see if the VRAM can recover from a write correctly. This is done to all sequential ascending locations. Next, a second independent pass of memory is made with the complementary data of `0xffffffff` written to descending locations of the FB memory buffer. The EMC mapping access modes are: • MDI_CHUNKY_XGBR_MAP • MDI_CHUNKY_BGR_MAP • MDI_PLANAR_X16_MAP • MDI_PLANAR_C16_MAP • MDI_PLANAR_X32_MAP • MDI_PLANAR_B32_MAP • MDI_PLANAR_G32_MAP • MDI_PLANAR_R32_MAP

TABLE 8-1 `cg14` Test Groups *(Continued)*

Test Groups	Description
MDI Chip Cursor Registers (Group 3)	The MDI Chip Cursor Registers are: • Cursor Plane 0 Register bits 31-0 write/read verify • Cursor Plane 1 Register bits 31-0 write/read verify • Cursor Plane 0 Register bits 31-0 write/read verify (with auto increment) • Cursor Plane 1 Register bits 31-0 write/read verify (with auto increment) • Cursor Control Register bits 2-0 write/read verify • Cursor Color Register 1 bits 28-0 write/read verify • Cursor Color Register 2 bits 28-0 write/read verify • X-Cursor Location Register bits 11-0 write/read verify • Y-Cursor Location Register bits 11-0 write/read verify • Cursor Plane 0 Non-Auto Registers test • Cursor Plane 0 Auto Registers test • Cursor Plane 1 Non-Auto Registers test • Cursor Plane 1 Auto Registers test • Cursor Planes Retry A test • Cursor Planes Retry B test
MDI Chip CLUT Registers (Group 4)	The MDI Chip CLUT Registers are: • LUT1 Registers 0-255 bits 31-27 & 23-0 write/read verify • LUT1 Registers 0-255 bits 31-27 & 23-0 write/read verify (with auto increment) • LUT1D Registers 0-255 bits 31-27 & 23-0 write/read verify • LUT1D Registers 0-255 bits 31-27 & 23-0 write/read verify (with auto increment) • LUT2 Registers 0-255 bits 31-27 & 23-0 write/read verify • LUT2 Registers 0-255 bits 31-27 & 23-0 write/read verify (with auto increment) • LUT2D Registers 0-255 bits 31-27 & 23-0 write/read verify • LUT2D Registers 0-255 bits 31-27 & 23-0 write/read verify (with auto increment) • LUT3 Registers 0-255 bits 31-27 & 23-0 write/read verify • LUT3 Registers 0-255 bits 31-27 & 23-0 write/read verify (with auto increment) • LUT3D Registers 0-255 bits 31-27 & 23-0 write/read verify • LUT3D Registers 0-255 bits 31-27 & 23-0 write/read verify (with auto increment)

TABLE 8-1 `cg14` Test Groups *(Continued)*

Test Groups	Description
DAC Chip Registers (Group 5)	The DAC Chip Registers test group includes the RAMDAC registers and control registers. RAMDAC Registers: • Address Register bits 7-0 (0x7 maximum) write/read verify • Mode Register bits 7-0 (skip bit 5) bits write/read verify Control Registers: • ID Register bits 7-0 r/o verify data is 0x8C • Pixel-Mask Register bits 7-0 write/read verify (skipped if dac rev= 2) • Command2 Register bits 7-0 write/read verify (skipped if dac rev = 2) • Command3 Register bits 7-0 write/read verify (skipped if dac rev = 2)
MDI Chip XLUT Registers (Group 6)	The MDI Chip XLUT Registers are: • XLUT Registers 0-255 bits 7-0 write/read verify • XLUT Registers 0-255 bits 7-0 write/read verify (with auto increment) • XLUTD Registers 0-255 bits 7-0 write/read verify • XLUTD Registers 0-255 bits 7-0 write/read verify (with auto increment)

TABLE 8-1 `cg14` Test Groups *(Continued)*

Test Groups	Description
CG14 Display (visual only) Group 7)	This test visually displays 256 boxes on the screen (each in a different color), and then shifts the CLUT1 entries giving the visual impression of the pattern mirroring itself from left to right horizontally. The pattern then rotates up, down, followed by mirroring itself horizontally left to right.
MDI Chip Test Mode Readback Register (Group 8)	This test mode reads back register bits 23-0 in read-only and verify modes.
Driver IOCTLs (Group 9)	• `MDI_GET_CFGINFO` check # of CLUT's, pixel height, pixel width, and pixel mode against hardware • `FBIOGATTR` check `real_type`, `fb_height`, `fb_width`, `fb_depth`, `fb_cmsize`, and `fb_size` against cfginfo values • `FBIOGTYPE` check `fb_type`, `fb_height`, `fb_width`, `fb_depth`,`fb_size`, and `fb_cmsize` against driver defines or cfginfo values • `FBIOGVIDEO` check status returned against hardware • `FBIOSVIDEO` set off, off, on, on, off verifying against hardware • `FBIOVERTICAL` (imbedded in `FBIOSVIDEO`) • `MDI_VRT_CNTL` turn off, off, on, on, off the video interrupt enable and verify the hardware agrees`MDI_SET_PIXELMODE` set different modes and verify against the hardware • `MDI_SET_PPR` set the different modes and verify against the hardware • `MDI_SET_COUNTERS` set HSS, HSC, XCC, HBC, XCS, HBS, CSC, VSS, VSC, VBC, VBS, HCT, and VCT then verify against hardware • `MDI_SET_XLUT` set xlut and verify against hardware • `MDI_GET_XLUT` get xlut and verify against hardware • `MDI_SET_CLUT` set clut (1-3 as applicable) and verify against hardware • `MDI_GET_CLUT` get clut (1-3 as applicable) and verify against hardware • `FBIOPUTCMAP` set and verify clut1 matches • `FBIOGETCMAP` verify clut1 matches get • `FBIOSATTR` set `emu_type` to `FBTYPE_MDICOLOR` and verify • `FBIOGATTR` check • `FBIOGCURMAX` verify `x` and `y` size match driver defines • `FBIOSCURSOR` verify set at 3 locations matches hardware • `FBIOGCURSOR` verify driver knows what set(s) just did • `FBIOSCURPOS` verify set at three locations matches hardware • `FBIOGCURPOS` verify driver knows what set(s) just did • `MDI_SET_CURSOR` set then check CCR, XCU, and YCU cursor hardware registers

TABLE 8-2 `cg14test` NTA Testing Patterns

NTA Test Pattern Number = *x*	*Test Data = y*	*New Data = z*
1.0	0x00000000	0x01010101
1.5	0x01010101	0xffffffff
2.1	0xffffffff	0xf1f1f1f1
2.2	0xf1f1f1f1	0x33333333
3.1	0x33333333	0xf0f0f0f0
3.2	0xf0f0f0f0	0x0f0f0f0f
4.1	0x0f0f0f0f	0x55555555
4.2	0x55555555	0xaaaaaaaa
5.1	0xaaaaaaaa	0x05050505 (1x) 0x88888888 (2x)
5.2	0x88888888	0xf5f5f5f5
6.1	0xf5f5f5f5	0x00000000 (1x) 0x5f5f5f5f (2x)
6.2	0x5f5f5f5f	0x11111111
7.1	0x11111111	0x00000000 (1x) 0xcccccccc (2x)
7.2	0xcccccccc	0xdbdbdbdb
8.1	0xdbdbdbdb	0x6d6d6d6d
8.2	0x6d6d6d6d	0x6b6b6b6b
9.1	0x6b6b6b6b	0x0000000
9.2	0x00000000	-

`cg14test` Options

To reach the 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 may not include the device appropriate to this test. Refer to the *SunVTS User's Guide* for more details.

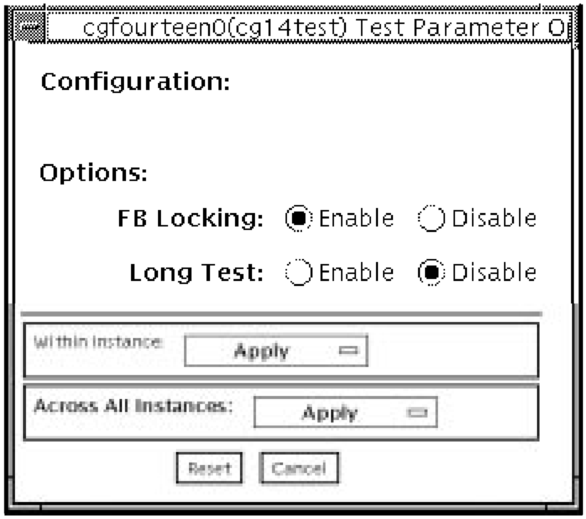


FIGURE 8-1 `cg14test` Test Parameter Options Dialog Box

TABLE 8-3 `cg14test` Options

Options	Description
FB Locking	See "Testing Frame Buffers" on page 9 for details.
Long Test	When enabled, the color bar screen(s) in the MDI Testmode Readback test checks all SAM transfers in clock=0 mode and clock=1 mode. If Long test is disabled, clock=1 runs checks on the first eight addresses and the first SAM transfer only.
Processor Affinity	For multiprocessor systems, indicates the processor to be tested.

`cg14test` Test Modes

Due to the nature of graphic tests, reading from or writing to the frame buffer during graphic tests will disturb user operation. This test is only available in offline Functional test mode.

TABLE 8-4 `cg14test` Test Modes

Test Mode	Supported?	Description
Connection	No	Not supported.
Functional (Offline)	Yes	This mode uses all subtests to test the cg14 frame buffer. The user can select the long mode for TRMB subtest.

`cg14test` Command-Line Syntax

`/opt/SUNWvts/bin/cg14test` *standard_arguments* **`-o dev=`***device_name*, **`lock=`***E(nable)/D(isable)*,**`L`**,**`I`**

TABLE 8-5 `cg14test` Command-Line Syntax

Argument	Description
`dev=`*device_name*	Specifies the path of the cg14 device file to be tested; for example: **`/dev/fbs/`***device_name.*
`lock=`*E(nable)/ D(isable)*	Enables and disables the window system locking option. See "Testing Frame Buffers" on page 9 for details. Do not use when your device is the window system display.
`L`	Enables the long TMRB test.
`I`	Enables optional driver ioctl tests for cursor. Note: Do not move the mouse during the `cg14test` when you run this option.

Note – 64-bit tests are located in the `sparcv9` subdirectory: `/opt/SUNWvts/bin/sparcv9/`*testname*. 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 "32-Bit and 64-Bit Tests" on page 3.

CHAPTER **9**

Frame Buffer, GX, GXplus, and TurboGX Options Test (cg6)

The cg6 test verifies the GX, GXplus, and TurboGX™ frame buffer and the graphics options offered with most SPARC™ based workstations and servers. This test stresses the frame buffer with the subtests described below.

For full instructions on testing frame buffers, see "Testing Frame Buffers" on page 9.

Note – Disable all screen savers before testing any graphics device. Type **xset s off** at the prompt to disable the Solaris screen saver.

To start SunVTS with vtsui, but without vtsk, add the host name to xhost as: **xhost +** *hostname*.

---

cg6 Subtests

TABLE 9-1 cg6 Subtests

Subtests	Description
Cursor test	cg6 visually checks the overlay registers of the RAMDAC. A pointer is drawn on the screen and moved to predetermined locations. There is a problem if the pointer disappears. This visual test ensures that the overlay is working properly
Fast Copy in double buffer test mode	Creates two full-size screen raster images in double buffer mode, writing different patterns to each. The hidden buffer is copied to the visible buffer, and the data is compared. An error message is returned if there are inconsistencies. Then the buffer is flipped and the process is repeated. Note: This test only applies to Sun Microsystems GX+ graphic accelerators with double-buffering capacity.
TEC test	Verifies that the transformation engine and cursor control logic are accessible to confirm that further TEC access is performed correctly.
FBC test	Verifies that the frame buffer controller logic is accessible to confirm that further FBC access is performed correctly.
Frame Buffer test	Verifies that the frame buffer memory is working. A “walking ones” pattern is written to memory, with a specific color signifying one of eight bits. The screen is divided into eight equally wide vertical stripes. A “walking one” is written to each stripe, causing eight iterations of these stripes. The value written is read back and checked. If the values do not match, an error is reported.
Screen test using blits	Draws blocks of color and performs blit transfers to other portions of the screen. First, the entire screen is drawn with cyan, then a black block is placed in the upper-left corner. This subtest blits this block on the upper-right, lower-right, and lower-left corners, then “or’s” the whole image.
Blit test	Draws a block of data and blit into a location at the bottom-right rectangle.

TABLE 9-1 `cg6` Subtests

Subtests	Description
Line test	Draws lines on the screen in different colors with different data values. The data is read back and compared with the expected values. An error is returned in the case of a mismatch.
Polygon test	Draws hourglass-shaped polygons on the screen, using the four vertices. After all the polygons are rendered in the video memory, they are read back and the data compared with expected values. If there is a mismatch, an error is displayed.
Colormap test	Loads all 256 locations in the color map with a greyscale, both backward and forward. This means decreasing values are loaded to all R, G, and B values. Note: If the system being tested has a monochrome or greyscale monitor, visual color problems are undetectable.

`cg6` Options

To reach the 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 may not include the device appropriate to this test. Refer to the *SunVTS User's Guide* for more details.

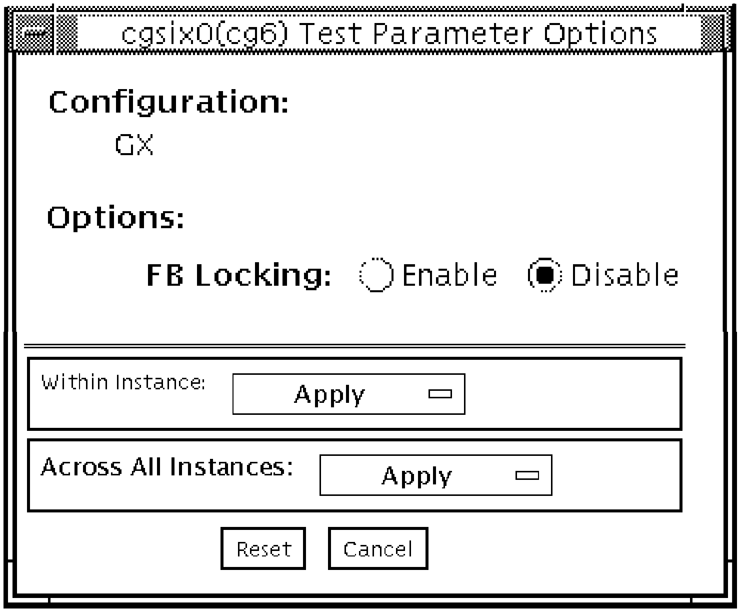


FIGURE 9-1 `cg6` Test Parameter Options Dialog Box

Note – See “Testing Frame Buffers” on page 9 for details about frame buffer locking.

`cg6` Test Modes

Due to the nature of graphic tests, reading from or writing to the frame buffer during graphic tests will disturb user operation.

TABLE 9-2 `cg6` Test Modes

Test Mode	Supported?	Description
Connection	No	Not supported
Functional (Offline)	Yes	Runs the full set of tests

`cg6` Command-Line Syntax

`/opt/SUNWvts/bin/cg6` *standard_arguments* **`-o dev=`***device_name***`,`** **`lock=`***E(nable)/ D(isable)***`,Passes=`***number*

TABLE 9-3 `cg6` Command-Line Syntax

Argument	Description
`dev=`*device_name*	Specifies the path of the cg14 device file to be tested, for example, `/dev/fbs/`*device_name*.
`lock=`*E(nable)/D(isable)*	Enables / disables the window system locking option. See “Testing Frame Buffers” on page 9 for details. Do not use when device is the window system display.
`Passes=`*number*	The number of passes to run. The default is 1.

Note – 64-bit tests are located in the `sparcv9` subdirectory: `/opt/SUNWvts/bin/sparcv9/`*testname*. 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 “32-Bit and 64-Bit Tests” on page 3.

Note – Extra swap space of 5 MBytes is required.

CHAPTER **10**

CPU Test (`cputest`)

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

The `cputest` comprises two subtests:

- g0 subtest—tests a processor's g0 register functionality. The g0 subtest is only supported on UltraSPARC-based systems.
- 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
  - `PnnnC`*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 72 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.

`cputest` Options

To reach the 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 may not include the device appropriate to this test. Refer to the *SunVTS User's Guide* for more details.

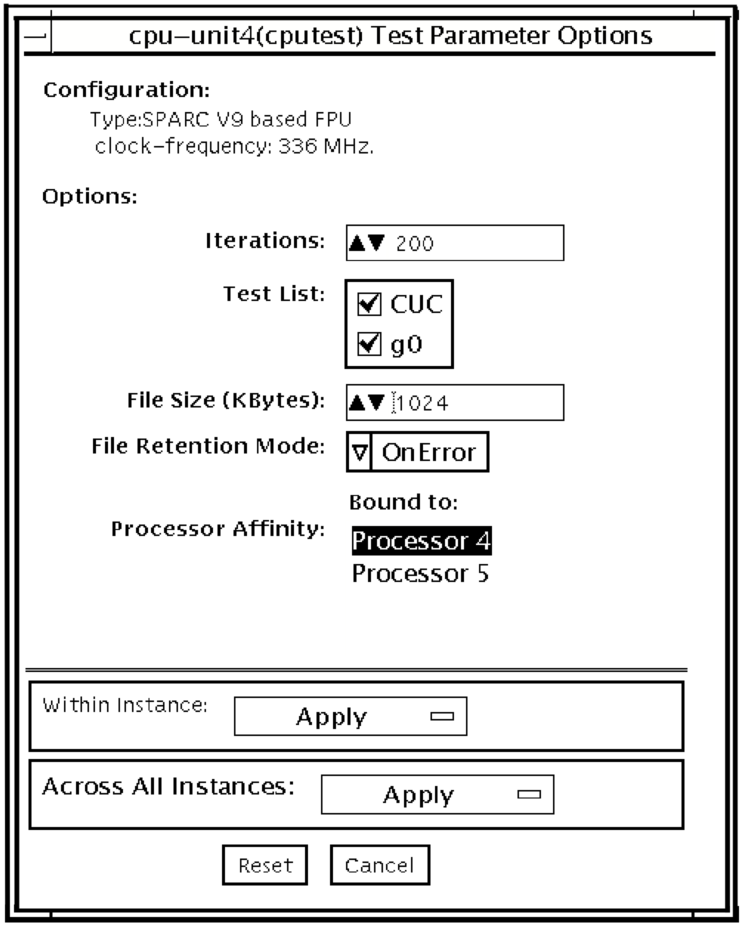


FIGURE 10-1 `cputest` Test Parameter Options Dialog Box

TABLE 10-1 `cputest` Option Dialog Box Descriptions

Option	Description
Iterations	Specifies the number of times to loop on the selected subtests. Use the up/down arrows 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. 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 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 10-2 `cputest` Test Modes

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

`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 10-3 `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 10-3 `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*. 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 “32-Bit and 64-Bit Tests” on page 3.

CHAPTER 11

CPU Power Management Test (cpupmtest)

cpupmtest cycles a CPU through its Power Management™ states. The CPU is run for user-defined periods of time from full speed, to half speed, to lowest speed, back up to half speed, and to full speed, in that order, at various levels. cpupmtest test verifies that the CPU speed changes correctly for each state.

You can also run this test concurrently with device tests, to monitor whether CPU speed changes are affecting device performance. Use cpupmtest to check that all devices function correctly during different Power Management modes.

Note – The number of speed levels available for testing depends on the type of CPU being tested, Enter an appropriate number of arguments for your CPU: speed1, speed2 .. speedn, where speed1 is the CPU's lowest speed and speedn is the CPU's highest speed.

cpupmtest is currently supported on Sun Blade™ 100 and Sun Blade 1000 systems.

Special Setup for cpupmtest

To run the cpupmtest properly, you must perform the following procedure:

Power Management Setup for `cpupmtest`

1. **Disable the Power Management autopm mode by making sure that the following lines are in the** `/etc/power.conf` **file:**

```
autoshutdown    30  9:00  9:00  noshutdown
autopm          disable
```

```
set us:us_direct_pm = 0x1
```

Note – You must reboot the system after making changes in the `/etc/power.conf` file.

`cpupmtest` Options

To reach the 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 may not include the device appropriate to this test. Refer to the *SunVTS User's Guide* for more details.

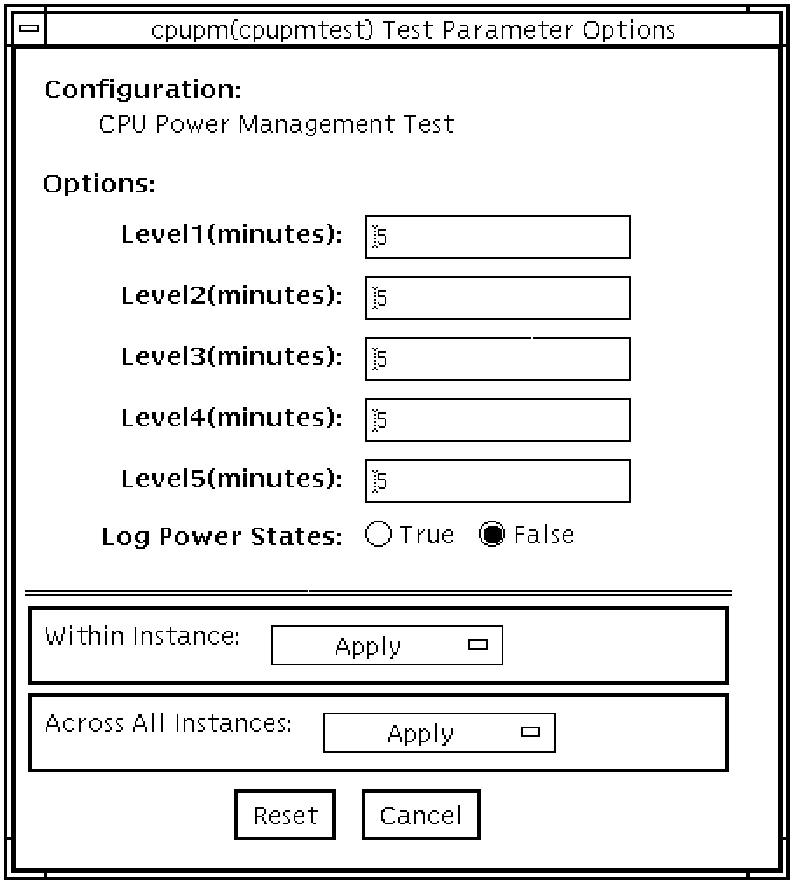


FIGURE 11-1 `cpupmtest` Test Parameter Options Dialog Box

Note – Your system may display a different number of levels in the dialog box, as the number of available levels depends on the type of CPU being tested.

TABLE 11-1 `cpupmtest` Options

Option	Description
Level1	Sets the number of minutes to test the system at its lowest speed.
Level2	Sets the number of minutes to test the system at a level higher than the lowest speed.
Level3	Sets the number of minutes to test the system at a level higher than Level2 speed.

TABLE 11-1 `cpupmtest` Options *(Continued)*

Option	Description
Level<n-1>	Sets the number of mnutes to test the system at a level lower than the normal speed.
Level<n>	Sets the number of mnutes to test the system at its normal speed.
Log Power States	Records and timestamps every CPU power level change to the VTS log file (`/var/opt/SUNWvts/logs/sunvts.info`).

`cpupmtest` Test Modes

TABLE 11-2 `cpupmtest` Test Modes

Test Mode	Supported?	Description
Connection	No	Not supported.
Functional (Offline)	Yes	Runs the full test.

`cpupmtest` Command-Line Syntax

`/opt/SUNWvts/bin/cpupmtest` *standard_arguments*
`-o dev=`*cpupm***`,speed1=`***mn***`,speed2=`***mn***`,speed`***n***`=`***mn*

TABLE 11-3 `cpupmtest` Command-Line Syntax

Argument	Description
dev=*cpupm*	Specifies the name of the device.
speed1=*mn*	Sets the number of minutes to test the system at its lowest speed.
speed2=*mn*	Sets the number of minutes to test the system at the next level of speed.
speed*n***=***mn*	Sets the number of minutes to test the system at its highest speed.

CHAPTER 12

Disk and Floppy Drives Test (`disktest`)

`disktest` verifies the functionality of hard drives and diskette drives using three subtests (see TABLE 12-1): Media, File System, and Asynchronous I/O. Most disk drives, such as SCSI disks, native or SCSI floppy disks, IPI, and so on, are supported. The type of drive being tested is displayed at the top of the Test Parameter Options dialog box.

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 (described below). 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 have SunVTS pre-mount all mountable partitions, set the environment variable `BYPASS_FS_PROBE` to 0 (zero) before starting SunVTS. Pre-mounting can be disabled by unsetting `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 it as superuser under the name `/disktest_c0t3d0s0`

Caution – If a power failure occurs while the Media subtest is running in WriteRead mode, disk data may be destroyed.

Caution – Running the Media subtest on a disk partition in the WriteRead mode may cause data corruption if the same partition is being used by other programs. Only select this mode when the system is offline (not used by any other users or programs).

`disktest` tests the floppy drive regardless of 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 may 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.

TABLE 12-1 describes the `disktest` subtests.

TABLE 12-1 `disktest` Subtests

Subtest	Description
Media subtest	Verifies disk media by writing data to and reading data from the disk. The Media subtest treats a disk as one large chunk of contiguous data. This is a scalable test that can run multiple copies in read/write mode on the same disk partition. To avoid data corruption, all simultaneous instances of `disktest` communicate through a shared memory service. This ensures that different copies of the media subtest do not overlay the same disk block at the same time. The Media subtest runs in two different modes as described below: SyncIO: The SyncIO media test creates a random offset for a partition from which the media testing starts. Starting from this offset, the read (Readonly mode) or the write/read (WriteRead mode) begins in a sequential fashion. The test continues until the specified percentage of media is covered. AsyncIO: The AsyncIO media test always starts from the first block of the partition under test and covers the area specified by the Media Coverage percentage. AsyncIO uses the read/write feature of the Solaris disk driver to exercise the disk. In Readonly mode, the test sends a maximum of four asynchronous read packets, each with a random size and a random offset into the selected partition. 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 is tested. In WriteRead mode, one write packet is issued in every four read packets as a spot check of the write operation. Before data is written to a particular location, data is backed up, write-verified, and restored to its original state.
File System subtest	Verifies the disk system's integrity. The File System subtest exercises the partition being tested to determine if it is mounted. If the partition is not already mounted or pre-mounted, then the test is blocked. The test opens two temporary files (of the size specified on `File System File Size`) and performs a Read/Write test.

`disktest` Test Options

To reach the 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 may not include the device appropriate to this test. Refer to the *SunVTS User's Guide* for more details.

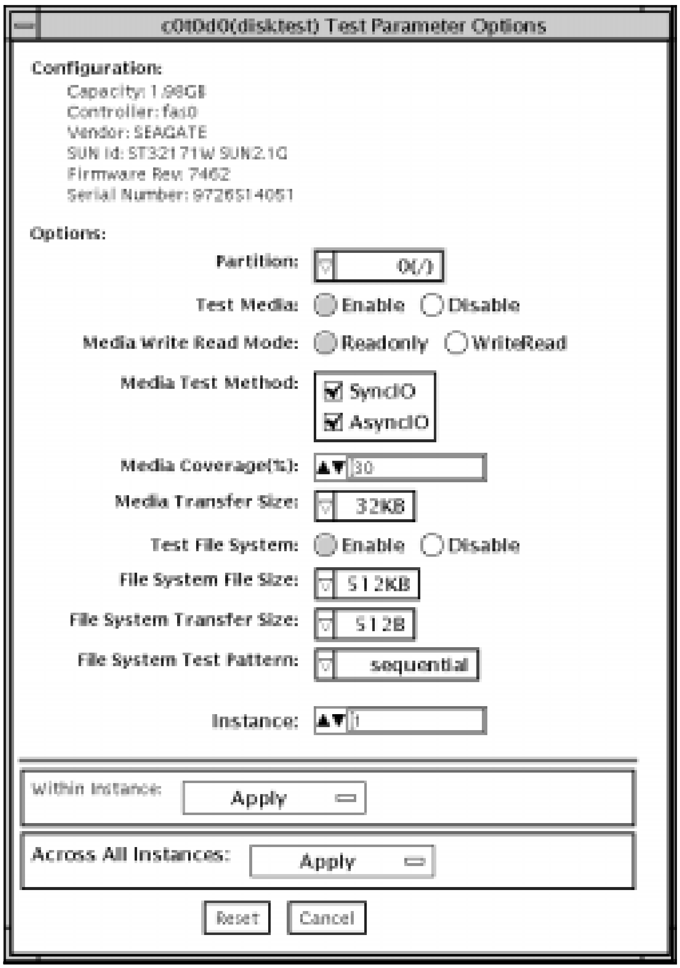


FIGURE 12-1 `disktest` Test Parameter Options Dialog Box

TABLE 12-2 describes the `disktest` option menu for different test modes.

TABLE 12-2 `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	Enables Read-Only or Read mode after write, with or without backup.
Media Test Method	Enables or disables the Media Test Methods (SyncIO and AsyncIO).
Media Coverage (%)	Enables users to test all or part of a partition (in percentages).
Media Transfer Size	Displays the transfer size of the media subtest.
Test File System	Enables or disables the File System subtest.
File System File Size	Creates a file system file size twice the size of what is specified.
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—[2KB] • Test File System—[Disable](fixed to Disable)

TABLE 12-2 `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(%)—[10] • Media Transfer Size—[2KB]] • Test File System—[Disable~](fixed to Disable)
Functional Test for Hard Disk	• Partition—0 - 7 [default] • Test Media—[Enable] [Disable] • Media Write Read Mode—[Read-only] [WriteRead] • Media Test method—[SyncIO] [AsyncIO] • Media Coverage(%)—[30] • 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]
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(%)—[30] • 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 12-3 disktest Test Modes

Test Mode	Supported?	Description
Connection	Yes	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.
Functional (Offline)	Yes	More than one instance of disktest is allowed for one disk device. The File System subtest, Media subtests, and floppy test can be run in offline Functional test mode.

disktest Command-Line Syntax

/opt/SUNWvts/bin/disktest *standard_arguments* **-o dev=***device_name* **,partition=***<0-7>***"(***mount_point***)",rawsub=***E | D***,rawrw=***Readonly | Writeread***, method=***AsyncIO+SyncIO***,rawcover=***n***,rawiosize=***n***,fssub=***E | D***,fssize=***n***,fsiosize=***n***,fspattern=***data_pattern*

TABLE 12-4 disktest Command-Line Syntax

Argument	Description
dev=*device_name*	Specifies the name of the disk to be tested, such as c0t3d0.
partition= *n***"(***mount_point***)"**	Specifies the partition number as follows: • *n*—is the partition number (slice number), usually *0-7* • *mount_point*—is 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.

TABLE 12-4 `disktest` Command-Line Syntax *(Continued)*

Argument	Description
`rawrw=` *Readonly | Writeread*	Specifies the Media subtest Read and Write mode: • Read only • Write, read then backup
`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
`rawcover=`*n*	Specifies media coverage from 0-100 (percentage) of the partition.
`rawiosize=`*n*	Specifies the media size to transfer. The number you specify is in kilobytes: `2KB|16KB|32KB|64KB|128KB|256KB|512KB`
`fssub=`*E(nable) | D(isable)*	Enables or disables the File System subtest.
`fssize=`*n*	Indicates the file system subtest size in kilobytes or megabytes: • `K|k|KB|kb`—kilobytes • `M|m|MB|mb`—megabytes `512KB|2MB| 8MB| 20MB|100MB|200MB`
`fsiosize=`*n*	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`
`fspattern=`*data_pattern*	Specifies the file system data pattern as sequential or random. {seq(uential) | 0x0(0000000) | 0xf(fffffff) | 0xa (5a5a5a5) | 0x5(a5a5a5a) | ran(dom) | 0xd(b6db6db)}

Note – 64-bit tests are located in the `sparcv9` subdirectory: `/opt/SUNWvts/bin/sparcv9/`*testname*. 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 "32-Bit and 64-Bit Tests" on page 3.

CHAPTER **13**

Sun Fire 880 FC-AL Disk Backplane Test (dpmtest)

dpmtest tests the Sun Fire™ 880 FC-AL disk backplane. dpmtest verifies the operation of the embedded SES controllers and the disk enclosure system of the Sun Fire 880 workgroup server.

No special hardware is required to run the dpmtest test.

dpmtest Options

To reach the 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 may not include the device appropriate to this test. Refer to the *SunVTS User's Guide* for more details.

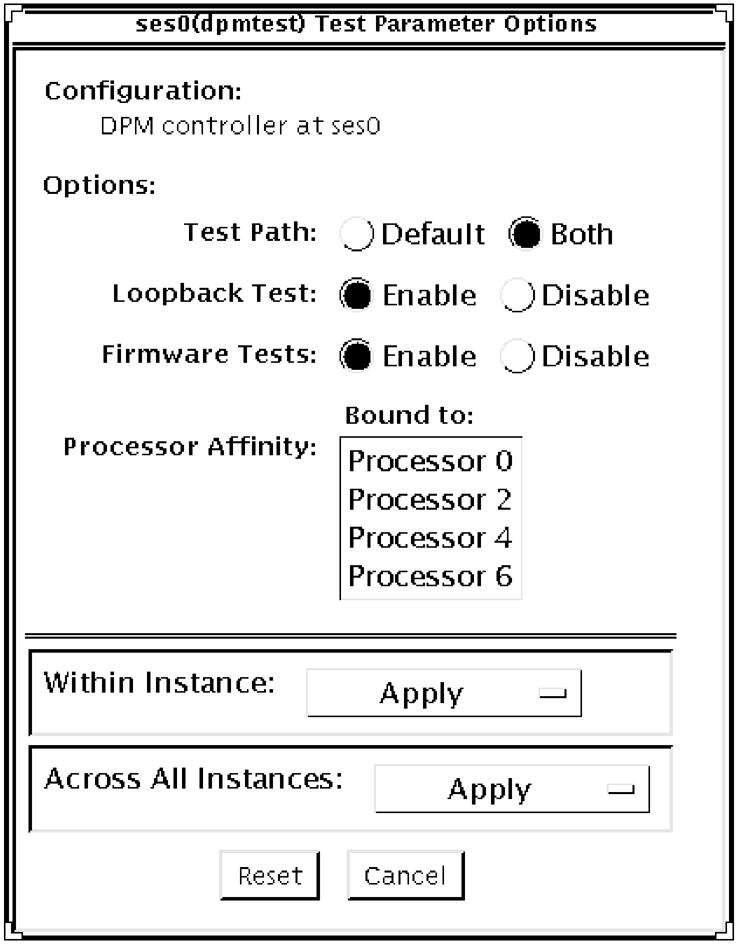


FIGURE 13-1 `dpmtest` Test Parameter Options Dialog Box

TABLE 13-1 `dpmtest` Test Options

Option	Description
Test Path	Default: Test runs using the interface through which the test was registered. If the device tested is an SES device, then the `dpmtest` tests the device over the fiber channel interface. If the device is registered as an SSC100 device, then it tests over the 12C interface. Both: If selected, the test runs the same tests to this device over both the Fiber channel interface and the 12C interface. Note: This option will be grayed out if both paths are not available.
Loopback Test	When enabled, the subtest will cause the SES device to loop a packet around the fiber bus with varying data patterns. The device reads the packet after the packet is received, and verifies that the data payload is correct. Note: This test grays out if the SSC100 does not have access to the fiber bus.
Firmware Test	When enabled, this subtest runs the system friendly embedded firmware tests on the SES/SSC100 device.

`dpmtest` Test Modes

TABLE 13-2 `dpmtest` Test Modes

Test Mode	Supported?	Description
Connection	Yes	The test opens the device, extracts information about the device (firmware revision, drives installed, temperatures, etc) and displays the information for the user. Once the test is run, the device closes and the test exits.
Functional (Offline)	Yes	The test opens the device and runs the selected tests against the device. When fully run, the test closes and reports the results.

`dpmtest` Command-Line Syntax

`/opt/SUNWvts/bin/dpmtest` *standard_arguments* **`-0 dev=`***[device name]***`,`**
`path=`*[Default | Both]***`,lb=`***[Enable | Disable]***`,fwtest=`***[Enable | Disable]*

TABLE 13-3 `dpmtest` Command-Line Syntax

Argument	Description
`-0 dev=` *[device name]*	*[device name]* is the path name of the device being tested.
`path=` *[Default | Both]*	Default: Test runs using the interface through which the test was registered. If the device tested is an SES device, then the `dpmtest` tests the device over fiber channel interface. If the device is registered as an SSC100 device, then it tests over the 12C interface. Both: If selected, the test runs the same tests to this device over both the Fiber channel interface and the 12C interface. Note: This option will be grayed out if both paths are not available.
`lb=`*[Enable | Disable]*	When enabled, the subtest will cause the SES device to loop a packet around the fiber bus with varying data patterns. The device reads the packet after the packet is received, and verifies that the data payload is correct. Note: This test will be grayed out in the case where the SSC100 does not have access to the fiber bus.
`fwtest=` *[Enable | Disable]*	When enabled, this subtest runs the system friendly embedded firmware tests on the SES/SSC100 device.

CHAPTER **14**

Sun StorEdge Hardware RAID Adapter Test (`dpttest`)

The `dpttest` exercises the Sun StorEdge™ PCI SCSI Hardware RAID adapter card. The `dpttest` can test this card with or without devices attached (see notes below).

The test runs firmware tests on the card to check the system interface as well as to verify functionality.

The `dpttest` performs the following subtests:

- DMA subtest—Transfers a 512-byte data block to the card, and then transfers it back. The test compares the result against the original data block and verifies the ECC code that is generated.
- BIST subtest—Re-runs the adapter's built in self test (BIST). BIST tests include:
  - Compare local RAM to flash RAM (quick RAM verify)
  - Data Path test to Domino RAM (cache)
  - Data path test to PLX module
  - Domino RAM test
- MEM-CHECK subtest—Checks the installed add-on memory (cache) to verify that the correct size and number of SIMMs are installed.

Note – If there are disks attached to the adapter, further diagnostic coverage can be obtained by running the `disktest` on any attached disks and/or RAID configurations.

Note – If the card has devices attached, some tests (the BIST RAM subtest) may not run. However, no error will be reported. This is the normal Functional test mode and allows the test to be run on a system that is currently in use.

---

dpttest Options

To reach the 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 may not include the device appropriate to this test. Refer to the *SunVTS User's Guide* for more details.

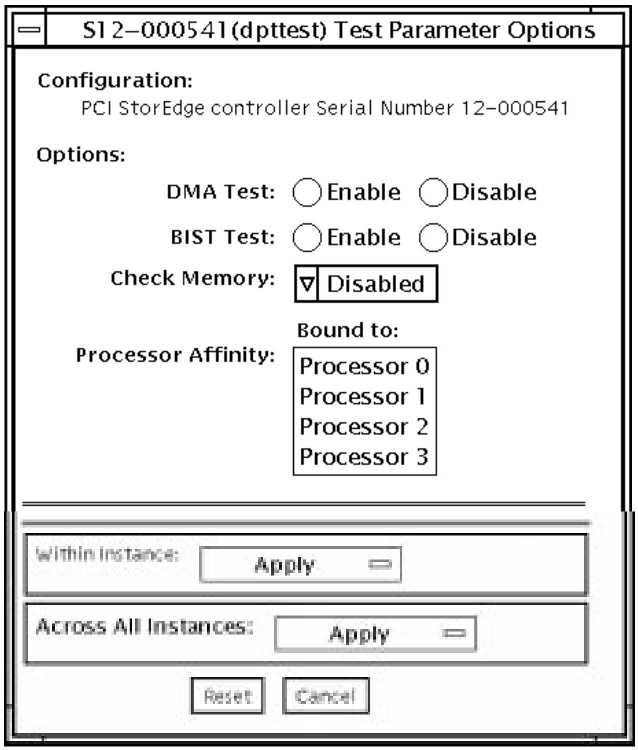


FIGURE 14-1 dpttest Test Parameter Options Dialog Box

TABLE 14-1 `dpttest` Options

`dpttest` **Options**	**Description**
DMA test	Enables or disables the DMA subtest. Default value—enabled for functional test mode, disabled for connection test mode.
BIST test	Enables or disables the BIST subtest. Default value—enabled for all test modes.
Check Memory	Enables or disables the memory subtest. Default—disabled for all test modes. Note: Sun currently supports 64 megabytes of cache on the SRC/P Hardware RAID card. To check the memory amount, set "Check Memory" to 64 M.

dpttest Test Modes

TABLE 14-2 dpttest Test Modes

Test Mode	Supported?	Description
Connection	Yes	The dpttest polls for the card, and runs only the BIST test.
Functional (Offline)	Yes	All subtests are run

dpttest Command-Line Syntax

/opt/SUNWvts/bin/dpttest [*standard_arguments*] **-o dev=***dev_serial_number***,**
dma=*Enable | Disable***,bist=***Enable | Disable***,**
memchk=*Disabled | 16M | 32M | 48M | 64M | 128M | 192M | 256M*

TABLE 14-3 dpttest Command-Line Syntax

Argument	Description
dev=*dev_serial_number*	Specifies the serial number of the target card to test. The serial number can be found on a label on the back side of the main card. Do not reference the serial numbers of the battery unit or the SCSI daughter card. There is no default value. This option must be specified.
dma=*Enable | Disable*	Enables or disables the DMA subtest. Default—Enabled for functional test mode, disabled for connection test mode.
bist=*Enable | Disable*	Enables or disables the BIST subtest. Default—Enabled for all test modes.
memchk=*Disabled | 16M | 32M | 48M | 64M | 128M | 192M | 256M*	Specifies the amount of memory to check with the Memory Check subtest. If disabled, no memory verification is done. If the value is set to one of the valid values, the test verifies the configuration of that amount of memory in the adapter. Default—Enabled for functional test mode, disabled for connection test mode. Note: Sun currently supports 64 megabytes of cache on the SRC/P Hardware RAID card. To check the memory amount, set "Check Memory" to 64 M.

Note – 64-bit tests are located in the `sparcv9` subdirectory: `/opt/SUNWvts/bin/sparcv9/`*testname*. 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 “32-Bit and 64-Bit Tests” on page 3.

CHAPTER 15

DVD Test (dvdtest)

The dvdtest tests the DVD by reading a DVD-ROM in the drive.

The dvdtest runs even if the Volume Manager is not running. If the Volume Manager is running and no media (DVD or CD) is installed in the drive, SunVTS prompts you to install media in the drive before selecting the test.

dvdtest Test Requirements

Note – Load a DVD-ROM into the DVD drive before running this test or the test will fail.

When a DVD-ROM is loaded in the drive, SunVTS uses the dvdtest to test the drive. When a CD (non-DVD type) is loaded, SunVTS uses the cdtest to test the drive. Whenever you change the media in the drive you must perform a reprobe (refer to the *SunVTS User's Guide* for details) so the SunVTS kernel will associate the correct test (dvdtest or cdtest) based on the media that is loaded in the drive.

dvdtest Options

To reach the 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 may not include the device appropriate to this test. Refer to the *SunVTS User's Guide* for more details.

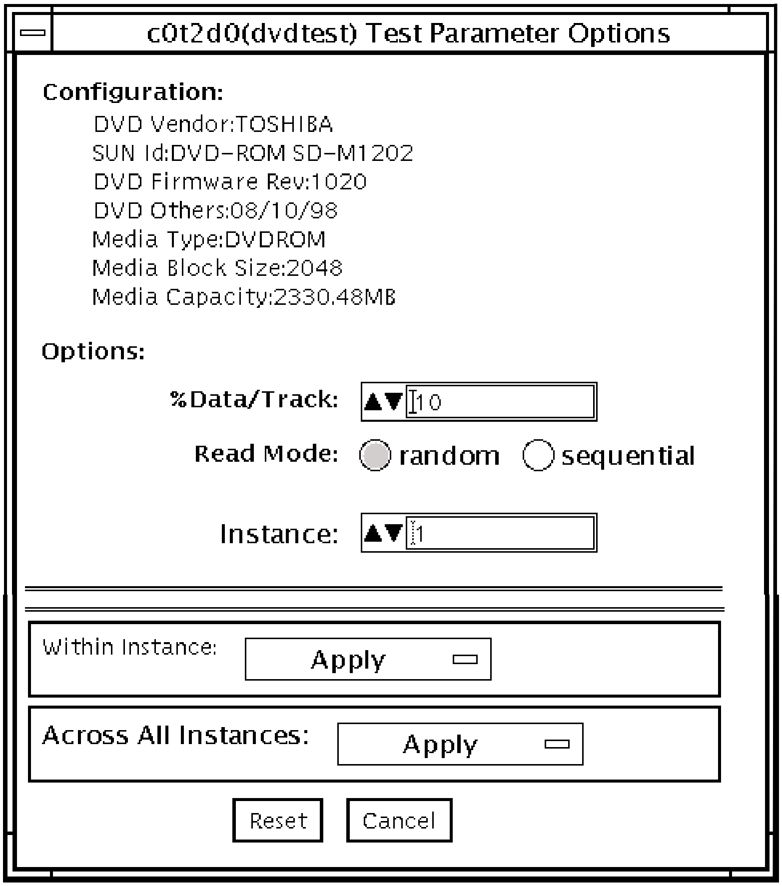


FIGURE 15-1 `dvdtest` Test Parameter Options Dialog Box

TABLE 15-1 `dvdtest` Option Menu Descriptions

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.

`dvdtest` Test Modes

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

TABLE 15-2 `dvdtest` Test Modes

Test Mode	Supported?	Description
Connection	Yes	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 (Offline)	Yes	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.

`dvdtest` Command-Line Syntax

`/opt/SUNWvts/bin/``dvdtest` *standard_arguments*
`-o dev=`*device_name***`,read=`**`random|sequential`**`,data=`***%_of_data*

TABLE 15-3 `dvdtest` Command-Line Syntax

Argument	Description
`dev=`*device_name*	Specifies the name of the device to test, for example `/dev/rdsk/c`*n*`t`*n*`d`*n*.
`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`.

Note – 64-bit tests are located in the `sparcv9` subdirectory: `/opt/SUNWvts/bin/sparcv9/`*testname*. 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 “32-Bit and 64-Bit Tests” on page 3.

CHAPTER **16**

ECP 1284 Parallel Port Printer Test (`ecpptest`)

The `ecpptest` verifies the functionality of the `ecpp`(7) IEEE 1284 parallel printer port device.

The `ecpp(7)` device is an *exclusive use device*. Only one application can interface with it at a time.

This test is not scalable.

`ecpptest` Hardware and Software Requirements

The `ecpp(7)` driver is installed with the operating system if the system being tested supports the `ecpp(7)` device. To run the optional printer subtest, a Centronics or ECP-mode printer must be attached to the printer port. To run the optional external loopback test, a passive loopback connector must be installed on the printer port.

Note – The external loopback test is intended for Sun internal manufacturing use only. It requires a custom loopback connector not available to an external customer.

`ecpptest` Subtests

TABLE 16-1 `ecpptest` Subtests

Subtest	Description
Internal Test FIFO Loopback	Verifies DMA and PIO accesses to the `ecpp` device. It uses an Internal Test FIFO on the `ecpp` device. There are no printer or loopback connectors required. This test is always executed.
External Passive Loopback	This verifies the parallel port I/O connections to the back panel connector. This test requires a passive loopback connector (Sun part no. 270-2965-01). This test is disabled by default and must be manually enabled by the user. Note: The external loopback test is intended for Sun internal manufacturing use only. It requires a custom loopback connector that is not available to an external customer.
Printer test	This verifies the parallel port printer operation. It outputs a half page of ASCII character data. The output mode (for example, ECP and Centronics) is determined by which mode the printer and `ecpp` driver automatically negotiate. It will not change the current mode. The user must verify that data printed properly. This test is disabled by default and must be manually enabled by the user.

`ecpptest` Options

To reach the 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 may not include the device appropriate to this test. Refer to the *SunVTS User's Guide* for more details.

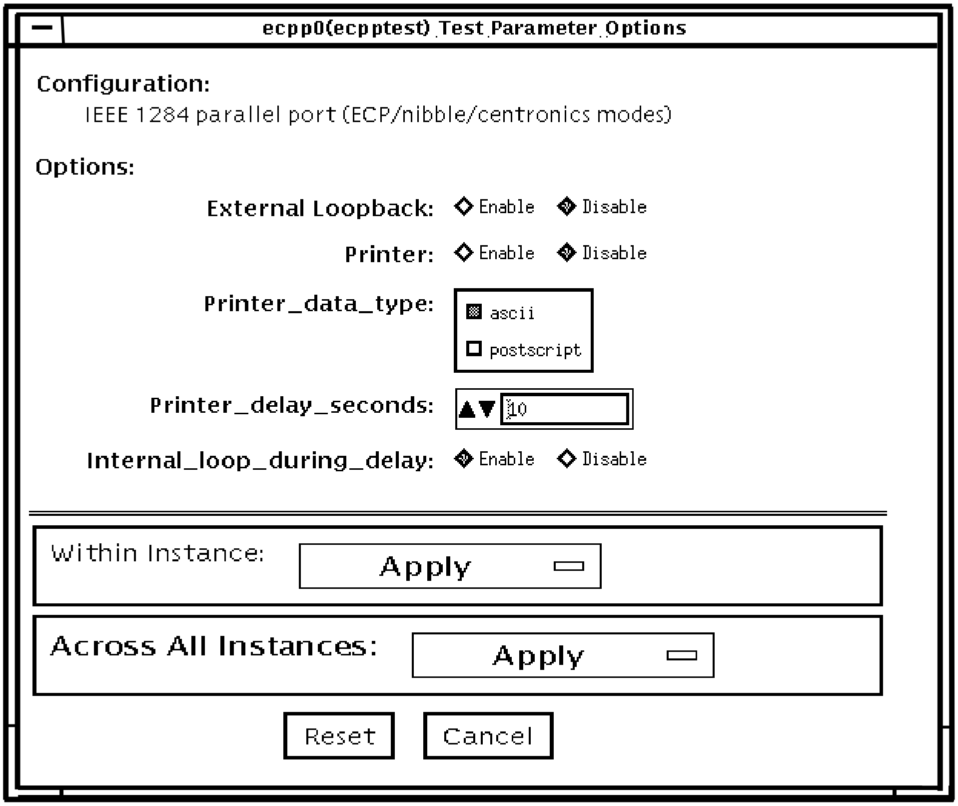


FIGURE 16-1 `ecpptest` Test Parameter Options Dialog Box

TABLE 16-2 `ecpptest` Options

`ecpptest` **Options**	**Description**
External Loopback	Enables or disables the external loopback test. This test requires a special loopback plug.
Printer	Enables or disables the printer test; this test requires a printer to be attached to the parallel port.

TABLE 16-2 `ecpptest` Options

`ecpptest` **Options**	**Description**
Printer_data_type	Choose whether ascii text or PostScript data is sent to the printer. The printer test must be enabled for this to be meaningful. A postscript printer must be attached to print postscript data.
Printer_delay _seconds	Allows the user to choose a delay between passes of the printer test. This prevents continuous printing of data that could quickly empty the paper supply. This is only meaningful if the printer test is enabled.
Interrnal_loop_ during_delay	Enables the Internal Test FIFO loopback test during the printer delay time. This is only meaningful if the printer test is enabled.

`ecpptest` Test Modes

TABLE 16-3 `ecpptest` Test Modes

Test Mode	Supported?	Description
Connection	Yes	Opens and closes the `ecpp`(7) device. No data is transferred. The test passes if the device can be open and closed successfully. The test also passes if the device cannot be opened because the device is busy with another process.
Functional (Offline)	Yes	Runs the internal loopback test and the user can optionally run the External loopback test and Printer test. The test will fail if the device is busy.

`ecpptest` Command-Line Syntax

/opt/SUNWvts/bin/ecpptest *standard_arguments* **-o [dev=***device_name***]** **[ext_loop=***Enable* | *Disable***][printer=***Enable* | *Disable***][,data=***ascii* | *postscript*][**,** **delay=***0-86400*][**,dloop=***Enable* | *Disable*]

TABLE 16-4 `ecpptest` Command-Line Syntax

Argument	Description
dev=*device_name*	Specifies the name of the device. This should be of the form `/dev/ecpp#,` where # is the minor number of the device. The default device is `/dev/ecpp0`.
ext_loop=*Enable* | *Disable*	If enabled, the external loopback test is run. The external loopback plug must be attached to the printer port.
printer=*Enable* | *Disable*	If enabled, the printer test is run. A Centronics or ECP mode parallel port printer must be attached
data=*ascii* | *postscript*	Choose whether ASCII text or PostScript data is sent to the printer. A PostScript printer must be attached to print postscript data.
delay=*0-86400*	Allows the user to choose a delay between passes of the printer test. This prevents continuous printing of data that could quickly empty the paper supply.
dloop=*Enable* | *Disable*	Enables the Internal Test FIFO loopback test during the printer delay time.

Note – 64-bit tests are located in the `sparcv9` subdirectory: `/opt/SUNWvts/bin/sparcv9/`*testname*. 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 “32-Bit and 64-Bit Tests” on page 3.

CHAPTER **17**

Sun StorEdge A5*x*00 Test (`enatest`)

`enatest` is used to provide configuration verification, fault isolation, and repair validation of the Sun StorEdge™ A5x00 subsystem. The `enatest` tests Sun StorEdge models A5000 (14 slot disk array) and A5200 (22 slot disk array).

The Sun StorEdge A5x00 is a high availability mass storage subsystem consisting of:

- SCSI Fibre Channel protocol host adapters with dual 100-Megabyte FC-AL ports.
- A disk enclosure.
- A front panel display for configuration information.
- Up to two interface boards in the enclosure, which provide FC-AL connections to the enclosure and also provide status information and control of the conditions within the enclosure.
- Other field-replaceable units (FRUs) within the enclosure include power supply units, fan trays, and backplane.

Note – Do not run `enatest` and `socaltest` at the same time, otherwise test failures might occur.

Note – The Sun StorEdge A5x00 was formally known as the Sun Enterprise Network Array™. The `enatest` tests both of these disk array subsystems.

---

enatest detects all Sun StorEdge A5x00 enclosures connected to the host and collects relevant configuration information. FIGURE 17-1 shows the Test Parameter Options menu which contains a sample configuration listing and test parameters. TABLE 17-1 describes the extent of the test coverage and provides samples of the configuration information that is displayed.

TABLE 17-1 enatest Coverage

Test Coverage	Description
Host Connections	enatest searches for all the active and inactive connections between the host and the enclosure and reports the number of existing active connections. If the VERBOSE mode is enabled, the port on the host side and the GBIC port on the enclosure side is reported for each active connection. The test also diagnoses any inactive connection(s) and reports the possible causes for the failure. The test will fail if there are one or more inactive connections. See the section on "enatest Fault Isolation Capability" on page 115 for more information.

TABLE 17-1 `enatest` Coverage

Test Coverage	Description

```
Sample Output, for an enclosure attached to an SBus socal card:

SUNWvts.enatest.1010 06/05/97 13:48:53 enatest ses0 VERBOSE:
“MYBOX: Lower-Right GBIC connected to host via /devices/sbus@1f,0/
SUNW,socal@0,0:1”
SUNWvts.enatest.1006 06/05/97 13:48:53 enatest ses0 VERBOSE:
“MYBOX: Interface Board (Bottom one in the enclosure) detected to be
installed
and OK”
SUNWvts.enatest.6023 06/05/97 13:48:53 enatest ses0
ERROR: “MYBOX: Cannot communicate with the enclosure via
/devices/sbus@1f,0/SUNW,socal@0,0:0; possibly connected to Lower-Left
GBIC in the enclosure”
Probable_Cause(s):
     (1)Signal too low at the GBIC module in the enclosure
     (2)Faulty cable or cable disconnected
     (3)Faulty GBIC module on the host side
Recommended_Action(s):
     (1)Ensure the cables are properly connected
     (2)Please contact your service representative

SUNWvts.enatest.2006 06/05/97 13:48:53 enatest ses0 INFO:
“MYBOX: Number of connections to the host: 1"
```

Disk Access	During the testing, each disk is accessed through each active connection leading to that disk. The `enatest` opens partition 2 on the disk and reads 512 bytes of raw data. If there are any failures, the test tries to isolate the fault to either an enclosure element, the cable, the host adapter card, or the OE module on the host adapter. See the section on “enatest Fault Isolation Capability” on page 115 for more information.
Enclosure Status	The status of the enclosure is obtained by querying the SCSI Enclosure Services (SES) device in the enclosure. Detailed information regarding the status of the elements within the enclosure is reported. The test fails if a critical condition is detected in the enclosure. The table below shows how the status information is reported.

TABLE 17-2 Element Enclosure Status

Enclosure Element	Information
Disk	• Fault Sensed—Yes / No • Status of ports A and B—Connected or Bypassed
Power Supply	• Status—ON / OFF • Temperature—OK / Critical Overtemp / Abnormal • AC Input—OK / Not OK • DC Output—OK / Not OK
Fan	• Status—On / Off • Speed—High / Low / Stopped
Backplane	• Status—OK / Failed • Status of ports A and B—Connected / Bypassed
Interface Board	• Temperature—OK / Critical Overtemp • Loop A status—OK / Failed • Loop B status—OK / Failed
GBIC	• Status—Disabled / Enabled • Signal Level—OK / Too low • Transmitter—OK / Failed

`enatest` Options

To reach the 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 may not include the device appropriate to this test. Refer to the *SunVTS User's Guide* for more details.

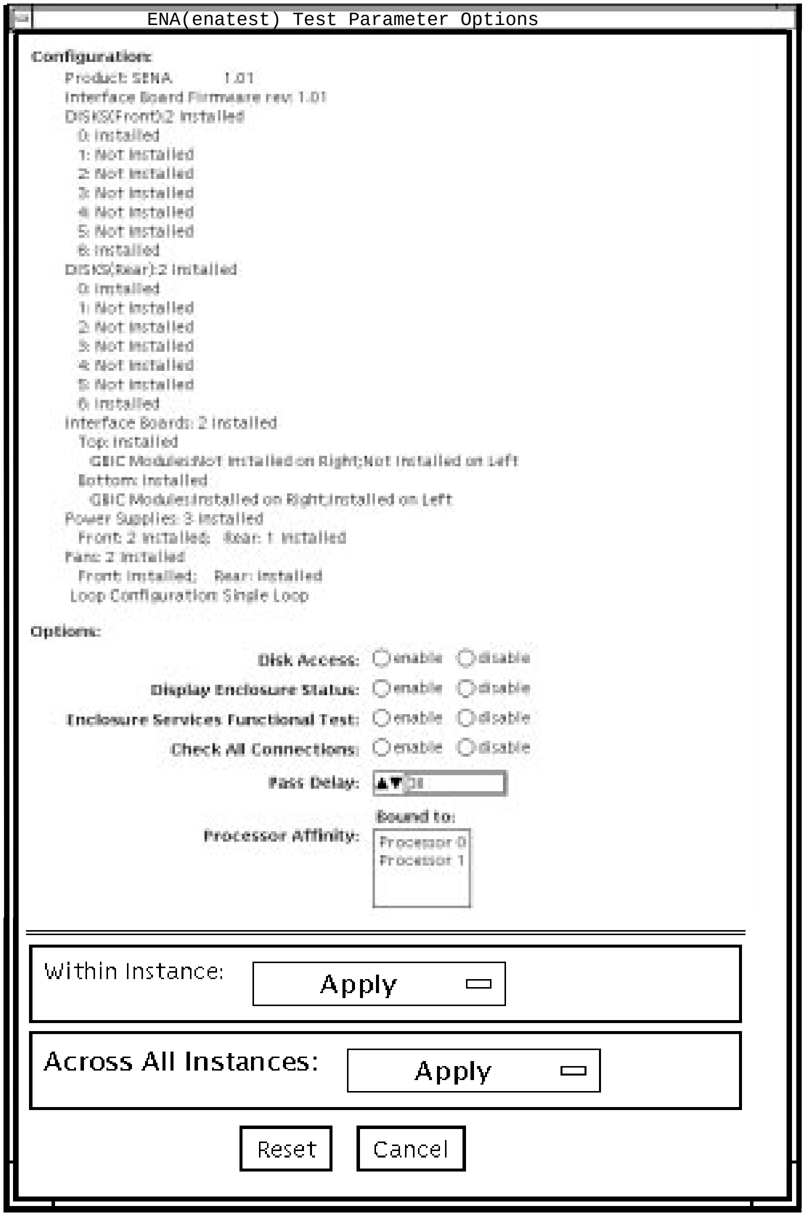


FIGURE 17-1 `enatest` Test Parameter Options Dialog Box

TABLE 17-3 `enatest` Options

`enatest` **Options**	**Description**
Enclosure Services Functional test (general description)	Certain control operations are performed on devices in the enclosure through the SES device and verified that the operation was performed successful. This functional test involves the following steps: 1. Perform control operation. 2. Verify control operation was successful. 3. Restore state to what it was before 1. 4. Verify restore operation was successful. The test will fail if any one of the above steps fails. This test targets the disks and the fans in the enclosure.
Enclosure Services Functional test (detailed Disk test description)	1. Control Operation—Each port of the disk is toggled from its original state. A port that was originally connected will be bypassed and vice-versa. 2. Verify Control Operation—This is done in two ways. First, the new status of the disk ports is verified by reading the status through the SES device. The test will fail if the status read back does not reflect the change. Next, disk access is attempted through the port that was originally connected but has now been bypassed. The test will fail if the access attempt is successful. 3. Restore State—The port states are restored to what they were before the Control Operation. 4. Verify Restore Operation—This is done in two ways. First the status of the disk ports is verified by reading the status through the SES device. The test will fail if the status read back does not reflect the change. Next, disk access is attempted through the port that have been reconnected. The test will fail if the access attempt is unsuccessful.
Enclosure Services Functional test (detailed Fan test description)	1. Control Operation—The speed of each fan is toggled. Possible speeds are HIGH and LOW. 2. Verify Control Operation—The status is read back through the SES device and the speeds are compared. Failure to vary the speed in this case will result in an INFO message indicating that the fan speed could not be changed but does not result in a test failure. This is because SES can ignore fan speed change requests if required because of existing ambient temperature conditions. 3. Restore State—Restore the fan speed to the original speed. 4. Verify Restore Operation—This is similar to the Verify Control Operation step above.

enatest Fault Isolation Capability

In the case of a failure, the test aids in fault isolation by reporting the possible cause(s) of failure. The fault isolation capability varies depending on the nature of the fault and the system configuration. `enatest` can detect and isolate hard faults. The following table shows the fault isolation capability for different configurations. A `Yes` indicates that fault isolation capability is available for that component in that configuration and a `No` indicates lack of fault isolation capability for that component in that configuration.

The following table is not applicable when using the PCI-based Fibre Channel card due to the card's lack of fault isolation capabilities.

TABLE 17-4 `enatest` Fault Isolation Configurations

Connections to Enclosure	System Architecture	SOC+ Host Adapter	Host Side GBIC or Cable	Enclosure Elements			
				Disk	Backplane	IB	GBIC
Multiple	sun4u	Yes	Yes	Yes	Yes	Yes	Yes
Multiple	sun4d	Yes	No	Yes	Yes	No	No
Single	sun4u	Yes	No	No	No	No	No
Single	sun4d	Yes	No	No	No	No	No

`enatest` Test Modes

TABLE 17-5 `enatest` Test Modes

<table>
<tr><th>Test Mode</th><th>Supported?</th><th>Description</th></tr>
<tr><td>Connection test</td><td>Yes</td><td>In this mode, the host connections and the status of the enclosure are checked. The test fails if there are any broken connections or if a critical enclosure condition is detected. Noncritical conditions result in a warning. A sample of the output follows.</td></tr>
<tr><td colspan="3"><pre>
Connection test starting....

    ses0
        Status: Connected
        Enclosure:
           Product Anemones Enterprise Network Array,
           Enclosure Name=MYBOX,
        Host Connections:
           Number of Active Connections=2,
        Enclosure State:
           Critical Conditions=None, Non-Critical Conditions=None

Connection test complete
</pre></td></tr>
<tr><td>Functional (offline)</td><td>Yes</td><td>All test options are allowed in this mode.</td></tr>
</table>

`enatest` Command-Line Syntax

`/opt/SUNWvts/bin/enatest` *standard_arguments* **`-o dev=`***device_name***`,`**
`disk_access=`*enable | disable***`,disks=`***disk1:disk2:disk3:...diskn***`,disp=`***enable | disable,*

esfunc=*enable* | *disable***,conn=***enable* | *disable***,delay=***delay_in_seconds*

TABLE 17-6 `enatest` Command-Line Syntax

Argument	Description
dev=*device_name*	Specifies the name of an `ses` device in the enclosure.
disk_access=*enable* | *disable*	Enables or disables disk access.
disks=*disk1:disk2:disk3:... diskn*	Lists the disks contained in this enclosure that the `disk_access` test should attempt to access. The disk names are separated by colons (:). If this option is disabled, the test will access all disks found in the enclosure.
disp=*enable* | *disable*	Enables or disables the display of detailed status information regarding the enclosure elements.
esfunc=*enable* | *disable*	Enables or disables the Enclosure Services Functional test.
conn=*enable* | *disable*	Displays information about the connection to the host.
delay=*delay_in_seconds*	Sets the minimum delay (in seconds) between successive invocations of the test.

Note – 64-bit tests are located in the `sparcv9` subdirectory: `/opt/SUNWvts/bin/sparcv9/`*testname*. 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 “32-Bit and 64-Bit Tests” on page 3.

CHAPTER **18**

Sun StorEdge 1000 Enclosure Test (`enctest`)

`enctest` tests the Sun StorEdge™ 1000 enclosures. The enclosure can support either 12 1" 4GB drives or 8 1.6" 9GB drives and have redundant power and cooling. Two enclosure models are available:

- Sun StorEdge A1000—Disk tray with the hardware RAID controller
- Sun StorEdge D1000—Disk tray without the hardware RAID controller

`enctest` can be used for validation, configuration verification, repair verification, and fault isolation of both models.

The `enctest` probe detects all the connected Sun StorEdge enclosures and displays the status of the various elements in the enclosure.

`enctest` sets the Sun StorEdge D1000 LEDs as follows:

TABLE 18-1 `enctest` Status LEDs

LED State	Indicated Condition
On	Unrecoverable or critical condition
Blinking	Non-critical condition
Off	Informational or no condition to report

Note – The `ses` driver must be present on the system with the Sun StorEdge enclosure(s). If the `ses` driver is not present, SunVTS will not detect the Sun StorEdge hardware. Use the following command to see if the `ses` driver is present:
`pkginfo SUNWses`
Install the package if necessary.

`enctest` reports the status of the various elements in the enclosure. An error is registered if an unrecoverable or critical condition is detected. Noncritical conditions are reported through warning messages. The table below describes the information that is reported for each enclosure element.

TABLE 18-2 `enctest` Enclosure Status Report

Enclosure Element	Information reported (per device slot)
Disk	Present / Not Present Failed / OK
Power supply	Present / Not Present Failed / OK
Fan	Present / Not Present Failed / OK
Temperature	OK / Over temperature
RPA cache Battery (StorEdge A1000 only)	OK / Low Charge

`enctest` Options

To reach the 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 may not include the device appropriate to this test. Refer to the *SunVTS User's Guide* for more details.

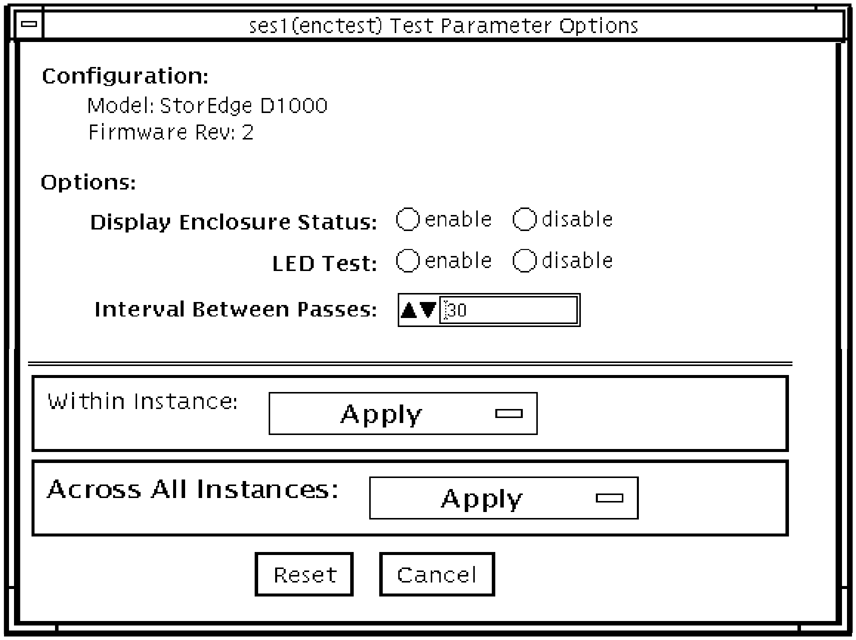


FIGURE 18-1 `enctest` Test Parameter Options Dialog Box

TABLE 18-3 `enctest` Options

`enctest` **Options**	**Description**
Display Enclosure Status	When enabled, the status of the enclosure elements are displayed as messages in the SunVTS console. These messages are also logged in the SunVTS info log file.
LED Test	When enabled, the LEDs associated with the disks are flashed for a short period of time.
Interval Between Passes	Sets the minimum delay between successive invocations of the test.

`enctest` Test Modes

TABLE 18-4 `enctest` Test Modes

Test Mode	Supported?	Description
Connectivity	Yes	The enclosure status summary bits are read. Detailed information is reported only if an unrecoverable or critical condition is detected.
Functional (Offline)	Yes	All options are allowed.

`enctest` Command-Line Syntax

`/opt/SUNWvts/bin/enctest` *standard_arguments* **`-o dev=`***device_name,* **`disp=`***enable | disable,***`led=`***enable | disable,***`delay=`***delay_in_seconds*

TABLE 18-5 `enctest` Command-Line Syntax

Argument	Description
`dev=`*device_name*	This is the name of a `ses` device in the enclosure.
`disp=`*enable | disable*	Detailed status information regarding the enclosure elements will be displayed if enabled.
`led=`*enable | disable*	Option to enable or disable the LED test.
`delay=`*delay_in_seconds*	Minimum delay between successive invocations of the test.

Note – 64-bit tests are located in the `sparcv9` subdirectory: `/opt/SUNWvts/bin/sparcv9/`*testname*. 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 "32-Bit and 64-Bit Tests" on page 3.

CHAPTER **19**

Environmental Test (`envtest`)

`envtest` exercises the I2C bus on the Sun Enterprise™ 450 product line. `envtest` contains five subtests to test and report on the power supply status, system temperature status, fan speed, disk LEDs, and front panel and keyswitch.

`envtest` is not scalable.

Note – Do not run `envtest` while the system is under a heavy load or false failures may be reported.

`envtest` Options

To reach the 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 may not include the device appropriate to this test. Refer to the *SunVTS User's Guide* for more details.

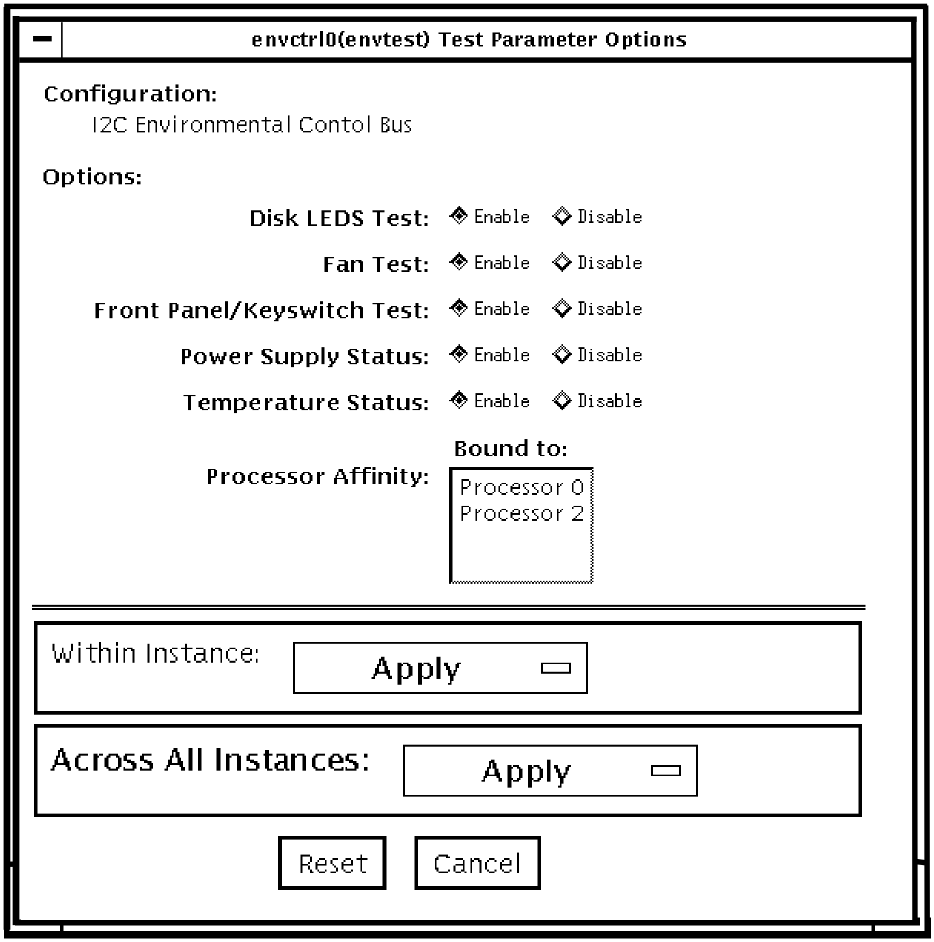


FIGURE 19-1 `envtest` Test Parameter Options Dialog Box

TABLE 19-1 `envtest` Options

`envtest` **Options**	**Description**
Disk LEDs test	Illuminates each LED on the disk backplane(s) to green, then amber, and then back to its original state. The test then illuminates all disk LEDs to green, then amber, and then back to their original state. This test is only enabled in Functional test mode.
Fan test	Cycles each fanbank speed to low, medium, and high, then verifies the correct speed. Next, each fanbank is stopped, one at a time. The test then verifies that a fan fault has occurred. Next, the watchdog timer is invoked to simulate a catastrophic failure. The test verifies that the system set all fanbanks to high and then resets the fan speed to normal. This test is only enabled in Functional test.
Front Panel and Keyswitch test	Flashes each individual LED on the front panel to ON (green or amber), then OFF, and then back to its original state. The test then illuminates all front panel LEDs then sets them back to their original state. The power on LED is Read Only and will not be cycled. The test then displays the current keyswitch position. This test is only enabled in Functional test mode.
Power Supply Status	Identifies the number of power supplies that are in the system, and the state of each power supply, and verifies that the power supply temperatures are within normal operating parameters. This test is enabled in all modes.
Temperature Status	Identifies the current temperature of each CPU in the system, and the ambient temperature of the system, and `envtest` verifies that all temperatures are within normal operating parameters. This test is enabled in all modes.

`envtest` Test Modes

`envtest` supports Connection and Functional tests.

TABLE 19-2 `envtest` Test Modes

Test Mode	Supported?	Description
Connection	Yes	Reports the status of the power supplies, the temperature sensors within the system, and verifies normal operating parameters.
Functional (Offline)	Yes	Tests the disk back panel, front panel LEDs, and fan control circuitry. Also uses the same functionality as online Functional mode and connection mode

`envtest` Command-Line Syntax

`/opt/SUNWvts/bin/envtest` [*standard arguments*]
`-o dev=`*raw_device_name*,**`diskleds=`***E/D*,**`fans`**=*E/D*,**`fpanel`**=*E/D*,
`psupply`=*E/D*,**`temp`**=*E/D*

TABLE 19-3 `envtest` Command-Line Syntax

Argument	Description
`dev=`*raw_device_name*	Specifies the name of the raw device to test.
`diskleds=`*enable | disable*	Enables or disables the Disk LEDS test.
`fans=`*enable | disable*	Enables or disables the Fans test.
`fpanel=`*enable | disable*	Enables or disables the Front Panel test.
`psupply=`*enable | disable*	Enables or disables the Power Supply test
`temp=`*enable | disable*	Enables or disables the Temperature test.

Note – 64-bit tests are located in the `sparcv9` subdirectory: `/opt/SUNWvts/bin/sparcv9/`*testname*. 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 “32-Bit and 64-Bit Tests” on page 3.

CHAPTER 20

Environmental Test (env2test)

env2test exercises and validates the I2C bus on the Sun Enterprise 250 systems.

Five subtests in env2test test and report the status of the power supply, system temperature, fan speed, disk LEDs, front panel, and keyswitch.

env2test is not scalable.

Note – Do not run env2test while the system is under a heavy load or false failures may be reported.

---

env2test Options

To reach the 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 may not include the device appropriate to this test. Refer to the *SunVTS User's Guide* for more details.

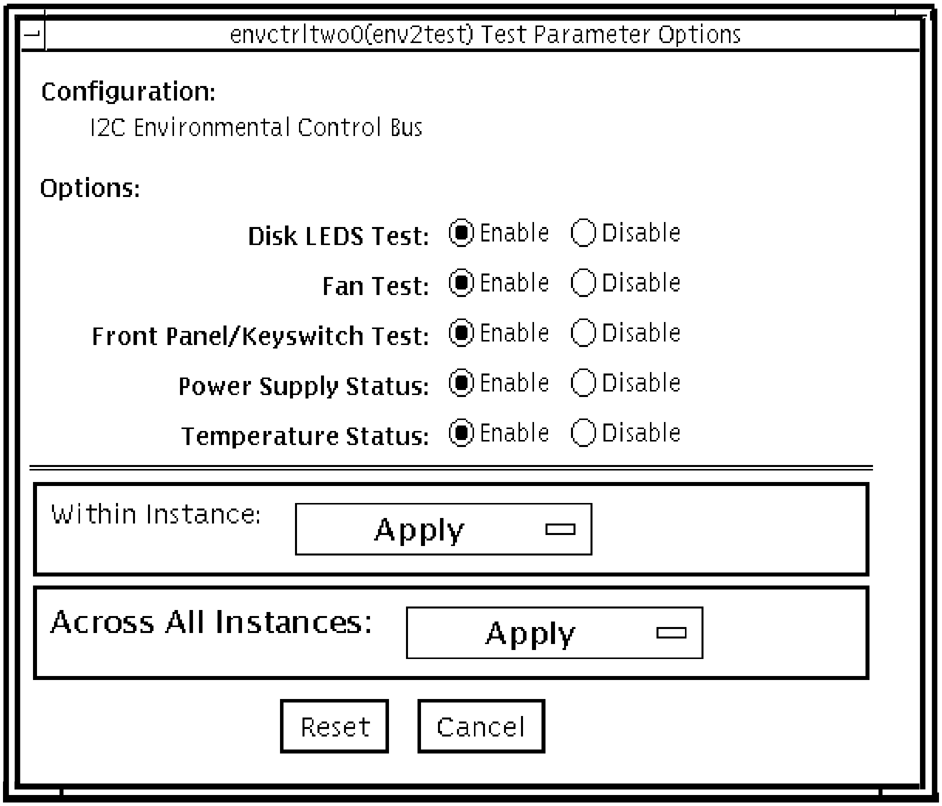


FIGURE 20-1 `env2test` Test Parameter Options Dialog Box

TABLE 20-1 `env2test` Options

`env2test` **Options**	**Description**
Disk LEDs test	Determines the number of disks that are present and the state of each disk. Illuminates each LED on the disk backplane to amber, and then back to its original state. The test then illuminates all disk LEDs to amber, and then back to their original state. This test is only enabled in Functional mode.
Fan test	Sets the fanbank speed to halfway between the current speed and max speed, then verifies the correct speed. Next, the fanbank speed is set to max speed, and the speed is verified. The fanbank speeds are then returned to normal, and then verified. This test is only enabled in Functional mode.
Front Panel and Keyswitch test	Flashes each individual LED on the front panel to On (green or amber), then Off, and then back to its original state. The test then illuminates all front panel LEDs then sets them back to their original state. The power on LED is read-only and will not be cycled. The test then displays the current keyswitch position. This test is only enabled in Functional mode.
Power Supply Status	Identifies the number of power supplies that are in the system and the state of each power supply. This test is enabled in all modes.
Temperature Status	Identifies the current temperature of each CPU in the system, the ambient temperatures of the system, the temperature on the SCSI and power distribution boards, and verifies that all temperatures are within normal operating parameters. This test is enabled in all modes.

env2test Test Modes

env2test supports Connection and Functional tests as described in the table below.

TABLE 20-2 env2test Test Modes

Test Mode	Supported?	Description
Connection	Yes	Reports the status of the power supplies and the temperature sensors within the system, and verifies normal operating parameters
Functional (Offline)	Yes	Tests the disk back panel, front panel LEDs, and fan control circuitry. Also uses the same functionality as online mode and connection test

env2test Command-Line Syntax

/opt/SUNWvts/bin/env2test [*standard arguments*]
-o dev=*device_name***,diskleds=***E/D***,env_mon=***poll_interval***,fans=***E/D*,
fpanel=*E/D***,psupply=***E/D***,temp=***E/D*

TABLE 20-3 env2test Command-Line Syntax

Argument	Description
dev=*raw_device_name*	Specifies the name of the raw device to test.
diskleds=*enable | disable*	Enables or disables diskleds test.
env_mon=*poll_interval*	Displays all system environmental statics every poll_interval seconds. (Display ONLY, does not test.)
fans=*enable | disable*	Enables or disables Fan test.
fpanel=*enable | disable*	Enables or disables Front Panel test.
psupply=*enable | disable*	Enables or disables Power Supply test.
temp=*enable | disable*	Enables or disables Temperature test.

Note – 64-bit tests are located in the `sparcv9` subdirectory: `/opt/SUNWvts/bin/sparcv9/`*testname*. 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 "32-Bit and 64-Bit Tests" on page 3.

CHAPTER **21**

Environmental Test (`env3test`)

`env3test` is an environmental control test for the Sun Blade 1000 and Sun Blade 100 systems. The test monitors the system by reading temperatures and fan speeds, as well as their limits. It reports whether the temperatures and fan speeds fall within system environmental condition limits.

The MAX1617 temperature sensor keeps a set of hard limits for the maximum and minimum temperatures allowed within the system. If the temperature passes one of these limits, the system performs a hard shutdown to protect hardware components. `env3test` also checks against the environmental monitor daemon, which keeps its own limits.

By reading these temperature values, the test reports the possibility of a failure of the system environmental feedback loop between the fans and the temperature sensors.

Note – If `env3test` fails to register temperature values, the system temperature indicators may be faulty.

Note – `env3test` will not run on operating environments earlier than the Solaris 8 10/00 operating environment.

`env3test` Options

To reach the 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 may not include the device appropriate to this test. Refer to the *SunVTS User's Guide* for more details.

FIGURE 21-1 `env3test` Test Parameter Options Dialog Box

If the Log File option is set to True, the test logs two lines of information read from the system into the log file `/var/opt/SUNWvts/logs/env3test.log`. The first line is a time stamp. The second line is a list of names and the read values, as shown below:

```
Wed May 24 13:55:57 2000
system-fan , 19, cpu-fan , 49, power-supply-fan , 100, cpu , 81,
cpu-ambient ,24
```

The values for fan settings refer to the percentage of performance at which each fan is running. The system fan, for example, is running at 19% of its capacity. The temperature values are in degrees Celsius. The cpu in the above example is running at 81 degrees C.

`env3test` Test Modes

TABLE 21-1 `env3test` Test Modes

Test Mode	Supported?	Description
Connection	Yes	Attempts connection to the device.
Functional (Offline)	Yes	Reports the received information to the GUI logging window in verbose mode.

`env3test` Command-Line Syntax

`/opt/SUNWvts/bin/env3test` [*standard arguments*] **`-o dev=`***driver_name***`,`** **`logging=`***true* | *false*

TABLE 21-2 `env3test` Command-Line Syntax

Argument	Description
`dev=`*driver_name*	Specifies the driver name for this test. The driver for the Sun Blade 1000 is `max1617`.
`logging=`*true* | *false*	Enables or disables the logging feature.

Note – 64-bit tests are located in the `sparcv9` subdirectory: `/opt/SUNWvts/bin/sparcv9/`*testname*. 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 “32-Bit and 64-Bit Tests” on page 3.

CHAPTER 22

Environmental Test (env4test)

env4test exercises and validates the I2C bus on the Sun Ct800 systems.

There are nineteen subtests in env4test that test and report the status of the following:

- Continuity LED flash test
- Slot LED flash test
- Disk LED flash test
- Power supply LED flash test
- Fan LED flash test
- System controller board LED flash test
- Front transition module LED flash test
- Green LED’s blink register
- Register read/write test, 3N march
- Register read/write test, 5N march
- Register read/write test, Crosstalk
- FRU presence test
- I/O slot health test
- I/O slot reset test
- Power remote on/off test
- Fan speed switch test
- CPU temperature status display test
- Power supply status display test
- Fan status display test

env4test is not scalable.

`env4test` Options

To reach the 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 may not include the device appropriate to this test. Refer to the *SunVTS User's Guide* for more details.

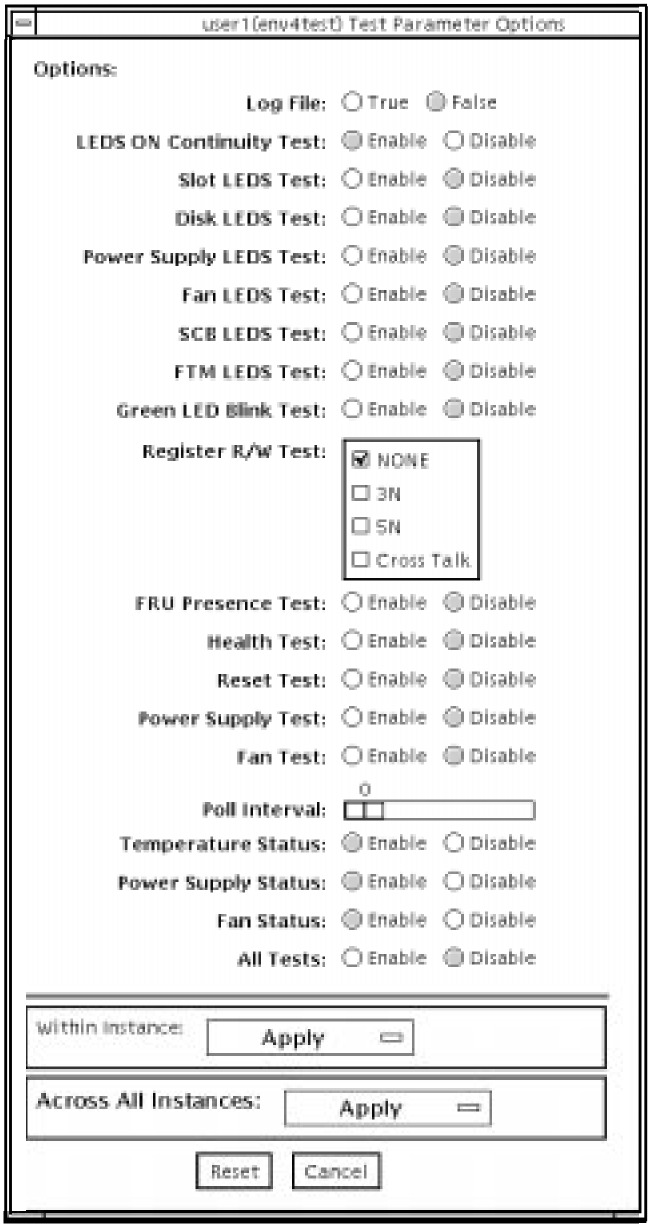


FIGURE 22-1 `env4test` Test Parameter Options Dialog Box

TABLE 22-1 `env4test` Options

`env4test` **Options**	**Description**
LEDS ON Continuity test	Asserts all LEDs in the system display panel except for the PDU LEDs. This verifies the ability of the System Controller Board (SCB) to perform sequential writes while automatically advancing the address pointer internally.
Slot LEDS test	Repeatedly flashes all I/O slot LEDs.
Disk LEDS test	Repeatedly flashes all disk LEDs.
Power Supply LEDS test	Repeatedly flashes all power supply LEDs.
Fan LEDS test	Repeatedly flashes all fan LEDs.
SCB LEDS test	Repeatedly flashes all system controller board LEDs.
FTM LEDS test	Repeatedly flashes all front transition module LEDs.
Green LED Blink test	Tests hardware functionality that enables blinking for the green LEDs.
Register R/W test	A pattern write test to interrupt registers and LED registers (read/write) to isolate the stuck-at-1, stuck-at-0, and cross talk fault categories. The test execution time from shortest to longest is: 3N march, 5N march, Cross Talk.
FRU Presence test	Reads and displays FRU presence signals.
Health test	Reads and displays I/O slot health signals.
Reset test	Sequentially resets individual I/O slots.
Power Supply test	Tests the remote power on/off signals in systems with redundant power supplies. This test is only valid when 2 power supplies are inserted in a single drawer.
Fan test	Tests the functionality of fan speed changes.
Poll interval	Displays all system environmental statics every poll_interval seconds.
Temperature Status test	Identifies the current temperature of the CPU, the ambient temperature of the system, and verifies that the temperature is within normal operating parameters. This test is enabled in all modes.

TABLE 22-1 `env4test` Options

`env4test` **Options**	**Description**
Power Supply Status test	Detects assertion of Presence signal in the SCB and reads the power supply status register via the I2C bus. The encoded bit status is displayed to the console.
Fan Status test	Detects assertion of Presence signal in the SCB and reads the fan status register via the I2C bus. The encoded bit status is displayed to the console.
All tests	Executes all tests sequentially.

`env4test` Test Modes

`env4test` supports Connection and Functional tests as described in the table below.

TABLE 22-2 `env4test` Test Modes

Test Mode	**Supported?**	**Description**
Connection	Yes	Reports the status of the power supplies, the temperature sensors within the system, and verifies normal operating parameters.
Functional (offline)	Yes	Tests the disk back panel, front panel LEDs, and fan control circuitry. Also uses the same functionality as Online mode and Connection test mode.

`env4test` Command-Line Syntax

`/opt/SUNWvts/bin/env4test` [*standard arguments*] **`-o dev=`***device_name*,
`ContinuityTest=`*E(nable)* | *D(isable)*,**`SlotLedsTest=`***E* | *D*,**`DiskLedsTest=`***E* | *D*,
`PsupplyLedsTest=`*E* | *D*,**`FanLedsTest=`***E* | *D*,**`ScbLedsTest=`***E* | *D*,
`FtmLedsTest=`*E* | *D*,**`GreenLedsBlinkTest=`***E* | *D*,**`RegTest=`***3N* | *5N* | *Cross_Talk*,

FRUPresenceTest=*E* | *D*,**HealthTest=***E* | *D*,**ResetTest=***E* | *D*,
PowerSupplyTest=*E* | *D*,**FanTest=***E* | *D*,**env_mon=***poll_interval*,**TempStatus=***E* | *D*,
PsupplyStatus=*E* | *D*,**FanStatus=***E* | *D*, **AllTests=***E* | *D*

TABLE 22-3 `env4test` Command-Line Syntax

Argument	Explanation
dev=*raw_device_name*	Specifies the name of the raw device to test.
ContinuityTest=*Enable* | *Disable*	Turns on all LED's except for PDU.
SlotLedsTest=*Enable* | *Disable*	Enables or disables the slot LED flash test.
DiskLedsTest=*Enable* | *Disable*	Enables or disables the disk LED flash test.
PsupplyLedsTest= *Enable* | *Disable*	Enables or disables the power supply LED flash test.
FanLedsTest=*Enable* | *Disable*	Enables or disables the fan LED flash test.
ScbLedsTest=*Enable* | *Disable*	Enables or disables the scb LED flash test.
FtmLedsTest=*Enable* | *Disable*	Enables or disables the ftm LED flash test.
GreenLedsBlinkTest= *Enable* | *Disable*	Enables or disables the green LED hardware blink test.
RegTest=*3N* | *5N* | *Cross_Talk*	Enables or disables the 3N, 5N, or Cross Talk register test.
FRUPresenceTest= *Enable* | *Disable*	Enables or disables the FRU presence test.
HealthTest= *Enable* | *Disable*	Enables or disables the health test.
ResetTest= *Enable* | *Disable*	Enables or disables the reset test.
PowerSupplyTest= *Enable* | *Disable*	Enables or disables the power supply remote on/off test.
FanTest=*Enable* | *Disable*	Enables or disables the fan speed change test.
env_mon=*poll_interval*	Displays all system environmental statics every `poll_interval` seconds. (Display ONLY, does not test.)
TempStatus=*Enable* | *Disable*	Enables or disables display of CPU temperature.
PsupplyStatus=*Enable* | *Disable*	Enables or disables display of power supply status.
FanStatus=*Enable* | *Disable*	Enables or disables display of fan status.
AllTests=*Enable* | *Disable*	Performs all tests in sequence.

Note – 64-bit tests are located in the `sparcv9` subdirectory: `/opt/SUNWvts/bin/sparcv9/`*testname*. 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 "32-Bit and 64-Bit Tests" on page 3.

CHAPTER 23

Environmental Test (env5test)

env5test exercises and validates the environmental subsytems of the Sunfire 280R product line. This test contains subtests to exercise a system's fans, keyswitch, LEDs, power supplies and temperature sensors.

This test is not scalable.

Note – Only the 64-bit version of this test is supported.

---

env5test Test Requirements

- You must install the SUNWpiclh, SUNWpiclr, SUNWpiclu, and SUNWpiclx picl packages correctly before running env5test.
- Verify that the picld daemon is running by typing the following:

```
# ps -ef |grep picld
root 100077      1  0   Sep 11 ?       23:40 /usr/lib/picl/picld
```

If the daemon is not running, run the script to restart it by typing the following:

```
# /etc/init.d/picld start
```

env5test Options

To reach the 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 may not include the device appropriate to this test. Refer to the *SunVTS User's Guide* for more details.

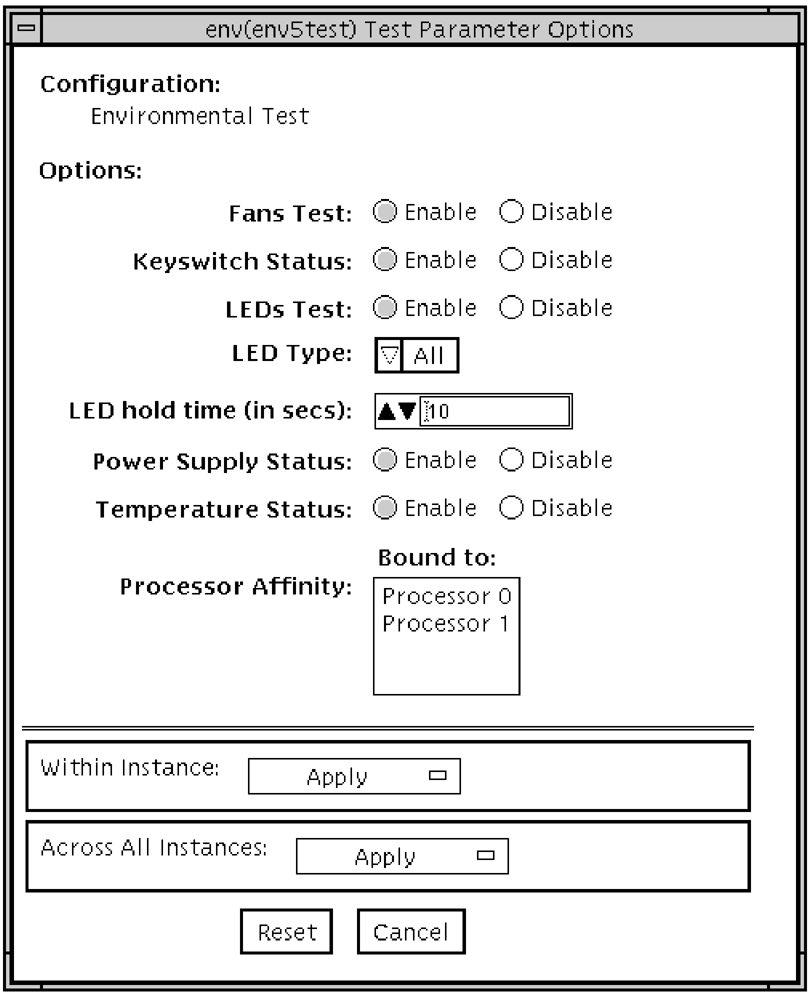


FIGURE 23-1 env5test Test Parameter Options Dialog Box

TABLE 23-1 `env5test` Subtests

Subtest	Description
Fan test	Cycles each fan bank in the system and identifies its current speed and state. Displays fault information if the state is not correct. Provides different test coverage for various types of fans based on their properties. Only enabled in Functional test mode.
Keyswitch status	Displays the current keyswitch position. Enabled in all modes.
LEDs test	Flashes each individual green or amber LED in the system on, then off, then back to its original state. You can select to test all LEDs (by default) or test by categories. Also specifies how long the LEDs stay on during the exercise. Only enabled in Functional test mode.
Power supply status	Identifies the number of power supplies in the system, the state of each one, and current (I) draw information if applicable. Displays fault information if the state of the power supply is incorrect. Enabled in all modes.
Temperature status	Identifies the current temperature of temperature sensors in the system and verifies that all temperatures are within normal operating parameters. Enabled in all modes.

`env5test` Test Modes

TABLE 23-2 `env5test` Test Modes

Test Mode	Supported?	Description
Connection	Yes	Runs the Keyswitch, Power Supply, and Temperature subtests. Reports on the status only of system fans. Verifies normal operating parameters.
Functional (Offline)	Yes	Runs the full set of subtests.

`env5test` Command-Line Syntax

`/opt/SUNWvts/bin/sparcv9/env5test` *standard_arguments*
`-o dev=`*raw_device_name***`,leds=`***E*|*D***`,ledtype=`***category***`,ledtime=`***num_seconds***`,`**
`keys=`*E*|*D***`,fans=`***E*|*D***`,psupply=`***E*|*D***`,temp=`***E*|*D*

TABLE 23-3 `env5test` Command-Line Syntax

Argument	Description
`dev=`*raw_device_name*	Specifies the name of the raw device to test. Default is `/dev/env`
`leds=`*E*|*D*	Enables or disables the LEDS subtest. Default is Disable.
`ledtype=`*category*	Sets the type of LED category to be tested. Default is All.
`ledtime=`*num_seconds*	Sets the number of seconds that LEDs are turned on during the LED subtest. Values are 0 to 10. Default is 0.
`keys=`*E*|*D*	Enables or disables the Keyswitch subtest. Default is Enable.
`fans=`*E*|*D*	Enables or disables the Fans subtest. Deafult is Disable.
`psupply=`*E*|*D*	Enables or disables the Power supply subtest. Default is Enable.
`temp=`*E*|*D*	Enables or disables the Temperature subtest. Default is Enable.

CHAPTER 24

Frame Buffer Test (`fbtest`)

`fbtest` is a generic test for all dumb frame buffers used with the Solaris 2.x and Solaris 7 software.

The `fbtest` checks the frame buffer by sequentially writing, reading, and verifying small blocks of random patterns across the entire video RAM. The block size is 64 x 64 pixels. If a miscompare occurs, the test stops with an error message that indicates the location of the error.

If a generic frame buffer device name (`dvc/fb`) is specified, `fbtest` automatically detects the depth of the frame buffer, and adjusts testing to the frame buffer size.

For full instructions on testing frame buffers, see "Testing Frame Buffers" on page 9.

`fbtest` Options

To reach the 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 may not include the device appropriate to this test. Refer to the *SunVTS User's Guide* for more details.

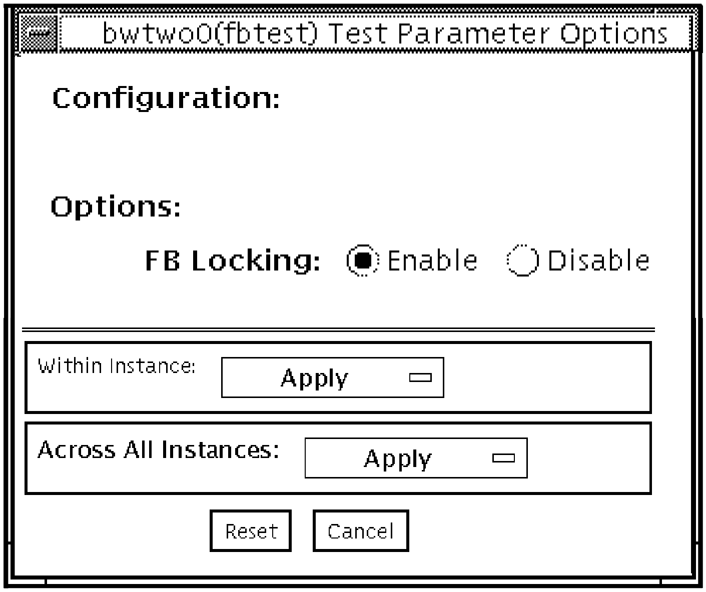


FIGURE 24-1 `fbtest` Test Parameter Options Dialog Box

Note – To start SunVTS with `vtsui`, but without `vtsk`, you must add the host name to `xhost` as: **`xhost +`** *hostname*.

`fbtest` Test Modes

Due to the nature of graphic tests, reading from or writing to the frame buffer during graphic tests disturbs user operation.

TABLE 24-1 `fbtest` Test Modes

Test Mode	Supported?	Description
Connection	No	Not supported
Functional (Offline)	Yes	Runs the full set of tests

`fbtest` Command-Line Syntax

`/opt/SUNWvts/bin/fbtest` *standard_arguments* **`-o dev=`***device_name*, **`lock=`***E(nable)/D(isable)*

TABLE 24-2 `fbtest` Command-Line Syntax

Argument	Description
`dev=`*device_name*	Specifies which frame buffer to test.
`lock=`*E(nable)/ D(isable)*	Enables or disables the window system locking option. See “Testing Frame Buffers” on page 9 for details. Frame buffer locking is enabled by default on the window server running the Open Windows software.

Note – 64-bit tests are located in the `sparcv9` subdirectory: `/opt/SUNWvts/bin/sparcv9/`*testname*. If a test is not present in this directory, then it may only be available as a 32-bit. For more information refer to “32-Bit and 64-Bit Tests” on page 3.

CHAPTER **25**

Fast Frame Buffer Test (`ffbtest`)

`ffbtest` verifies the functionality of the fast frame buffer (FFB).

`ffbtest` can detect and adapt to the video modes of single- and double-buffer versions of the FFB. All `ffbtest` tests can run in several screen resolutions such as standard, stereo, and high resolution. In stereo mode, all tests write into the right and left eyes unless you specify otherwise. This test also supports FFB, FFB2, and FFB2+ fast frame buffer configurations. Use the `ffbconfig -prconf` command to display the configuration of the frame buffer you want to test.

You can interrupt `ffbtest` using Control-C. Turn off all other keyboard input if OPEN LOOK is running on the unit being tested.

Test accuracy is checked using a checksum algorithm. Possible locations of failing pixels are identified, as well as the likely failing FRU.

Caution – *Do not* run any other application or screen saver program that uses the FFB accelerator port while running `ffbtest`. These programs cause SunVTS to return incorrect errors.

Note – *Do not* run Open Windows™ across multiple monitors or the test could fail.

`ffbtest` Test Requirements

Disable all screen savers before testing any graphics device. Type **`xset s off`** at a UNIX prompt to disable the Solaris screen saver. Disable the Power Management™ software if it is running.

For full instructions on testing frame buffers, see "Testing Frame Buffers" on page 9.

ffbtest requires approximately 7 Mbytes of disk space in the /tmp directory to extract its working files. If this space is not available, the diagnostic will fail and report warning and error messages, indicating a lack of disk space.

To start SunVTS with vtsui, but without vtsk, you must add the host name to xhost as: **xhost +** <*hostname*>.

ffbtest Options

To reach the 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 may not include the device appropriate to this test. Refer to the *SunVTS User's Guide* for more details.

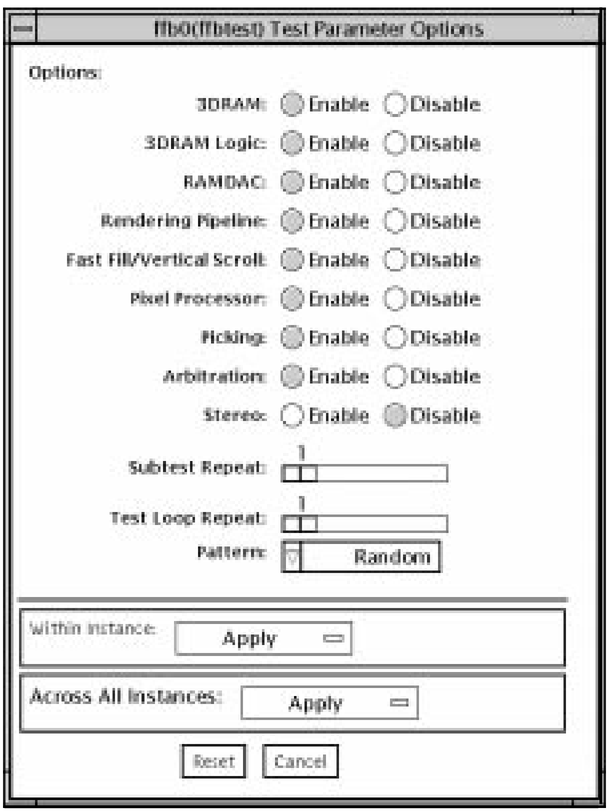


FIGURE 25-1 ffbtest Test Parameter Options Dialog Box

By default, all tests are enabled except the stereo test.

TABLE 25-1 `ffbtest` Options

`ffbtest` **Options**	**Description**
3DRAM test	The 3DRAM test thoroughly tests the video memory in the FFB using 512-bit reads and writes. 3DRAM makes a full screen pass, consisting of a write and a read to each pixel location, for each access mode on the list below. You can use either random data or specify data at the command line. A second pass is made with the one's complement of the data used in the first pass so that each memory location is tested with both a zero and a one. Notice that some passes are skipped on the single-buffered FFB. Errors in this subtest are attributes to the 3DRAM. A failing chip is indicated by (X, Y) locations and device-specific "U" numbers: • DFB8R, DFB8G, DFB8B, DFB8X—Buffer A • DFB24—Buffer A • DFB32—Buffer A • SFB8R, SFB8G, SFB8B, SFB8X—Buffer A • SFB8R, SFB8G, SFB8B, SFB8X—Buffer B (double buffer only) • SFB32—Buffer A • SFB32—Buffer B (double buffer only) • SFB32—Buffer C (double buffer only) • SFB64—Buffers A and C (double buffer only) • SFB64—Buffers B and C (double buffer only)
3DRAM Logic test	3DRAM Logic provides logical functionality to the FFB. The following services are tested: • Compare Controls—Match AB • Compare Controls—Magnitude AB • Compare Controls—Match C (double buffer only) • Compare Controls—Magnitude C (double buffer only) • Match Mask—AB • Magnitude Mask—AB • Match Mask—C (double buffer only) • Magnitude Mask—C (double buffer only) • Raster Operations—RGB • Raster Operations—X • Raster Operations—YZ (double buffer only) • Plane Mask—RGB • Plane Mask—X • Plane Mask—Y • Plane Mask—Z • Group Enable—R, G, B, X • Group Enable—Y, Z (double buffer only)

TABLE 25-1 `ffbtest` Options *(Continued)*

`ffbtest` **Options**	**Description**
3DRAM Logic test (Continued)	`ffbtest` tests each function separately with a series of SFB64 writes. A total of 16 writes are made for each different test case with Y coordinate values varying from 0 to 30 in increments of 2 pixels. This dotted column organization provides page thrashing and block flashing in all screen resolutions. For each operation, all possible combinations are tested. For example, `ROP RGB new==old` has three possible values: `new < old`, `new == old`, *and* `new > old`. `ffbtest` tests each of these cases. Five passes of the functions are made. Each pass writes into a different FFB address space: SFB32-A, SFB32-B, SFB32-C, SFB64-AC, and SFB64-BC. Note that the passes that write into the SFB32 address spaces are writing two pixels at a time because the tests use SFB64 writes. For FFB2+ boards, additional testing is performed on the new stencil and passin capabilities if the board is DBZ. Care is taken to ensure that all 3DRAM chips are tested. Errors in this subtest are attributed to the 3DRAM.
RAMDAC Test	RAMDAC registers are tested using simple read / write patterns to determine if there are any bad bits. This includes all LUTs. `ffbtest` ensures that data is actually being read from the RAMDAC and not being supplied by the driver. Next, the RAMDAC Signature Register captures the pixels going to the screen. This test determines that all of the different data paths within the RAMDAC are functioning properly. The following modes are tested: • 24-bit true color from A • 24-bit true linear color from A • 24-bit direct color from A • 24-bit true color from B (double buffer only) • 24-bit true linear color from B • 24-bit direct color from B (double buffer only) • 8-bit pseudo color (from each plane in RGB) from A • 8-bit pseudo color (from each plane in RGB) from B (double buffer only) • 8-bit non-linear grayscale (from each plane in RGB) from A • 8-bit non-linear grayscale (from each plane in RGB) from B (double buffer only) • 8-bit linear grayscale (from each plane in XRGB) from A • 8-bit linear grayscale (from each plane in XRGB) from B (double buffer only) • 8-bit overlay pseudo color (from buffer A, X plane)

TABLE 25-1 `ffbtest` Options *(Continued)*

`ffbtest` **Options**	**Description**
RAMDAC test (Continued)	This test displays a total of 11 different types of windows on the screen for the single-buffered configuration; 22 for double-buffered. A cursor is also displayed on the screen. RAMDAC on FFB2+ board supports three modes (SEP8, SEP4, and Combined). This test detects the RAMDAC type and tests the original and additional features like increased number of CLUTs, increased WLUT size, additional overlay WLUT. Errors in this test are attributed to the RAMDAC.
Rendering Pipeline test	Rendering Pipeline uses the rendering pipeline tests developed for the FFB stand-alone diagnostics. Each primitive is tested thoroughly with the following sources and configurations: • Dots • Anti-aliased dots • Lines using all four line drawing primitives • Triangles • Polygons • Rectangles • Fonts Errors in this test are attributed to the FBC.
Fast Fill / Vertical Scroll test	The Fast Fill / Vertical Scroll primitives are separated from the Rendering Pipeline tests because of their dependence on screen type. There are three different tests, one for each screen type. Each test uses both block and page mode fistfuls. Errors in this test are attributed to the FBC.
Pixel Process test	Pixel Processor, a subtest, exercises the following options selected by the FFB’s Pixel Processor Control (PPC) register: • Auxiliary clipping (additive and subtractive) • Depth cueing • Alpha blend • Viewport clip (2D and 3D) • Area pattern (transparent and opaque) Errors in this test are attributed to the FBC.

TABLE 25-1 `ffbtest` Options *(Continued)*

`ffbtest` **Options**	**Description**
Picking test	The Picking test exercises the pick detect login of the 3DRAM. `ffbtest` defines a pick detect window and checks that writes to the window are picked, and writes outside the window are not picked. The test is repeated once for each 3DRAM. Errors in this test are attributed to the 3DRAM.
Arbitration test	The Arbitration subtest continuously renders an object into the accelerator port while performing reads and writes through the direct port. For single-buffered configurations, a picture is rendered into the RGB planes while another process does DFB reads and writes in the X plane. For doubled buffered configuration, a picture is rendered into all 32 planes of the B buffer while the other does 32-bit DFB reads and writes in the A plane. This subtest simulates conditions in the real world, where rendering processes and windows operations run concurrently. Errors in this test are attributed to the FBC.

TABLE 25-1 `ffbtest` Options *(Continued)*

`ffbtest` **Options**	**Description**
Stereo Test	The Stereo test displays an object in stereo mode with different images for the right and left eye. The user can verify proper operation by looking at the screen with stereo glasses and following the displayed instructions. If the monitor type is not 1280x1024 at 76MHz, this test prints a warning message and does not run. To prevent this message from being displayed or written to the SunVTS information log, disable the stereo test in the test option menu. Only Sony P4 and N2 monitors support stereo resolutions. This test temporarily switches the monitor into stereo mode, renders a stereo image, performs a signature analysis on the stereo image (using the RAMDAC signature capture register), and after displaying the image for five seconds, restores the monitor to its previous resolution. Errors in this test are attributed to the RAMDAC. Note — If vertical lines are displayed on the console when running SunVTS, this could be caused by the `ffbtest` stereo test. There is a time-critical period in the FFB hardware when trying to change the screen resolution from standard to stereo and back to standard. When the system is heavily loaded or running all of the SunVTS tests, the FFB device driver may get interrupted while changing screen resolution. If this occurs, FB ASIC and RAMDAC get out of synchronization, resulting in an unusual display on the FFB screen. This problem could also cause a system hang condition. To avoid this type of display problem, disable the `ffbtest` stereo test when other SunVTS tests are enabled. This test is disabled by default because it is only needed when a stereo monitor and stereo glasses are present.

`ffbtest` Test Modes

Due to the nature of graphic tests, reading from or writing to the frame buffer during graphic tests will disturb user operation. This test is only available in offline Functional test and Stress mode.

TABLE 25-2 `ffbtest` Test Modes

Test Mode	Supported?	Description
Connection	No	Not supported
Functional (Offline)	Yes	The `ffbtest` verifies both the single- (SFB) and double-buffered (DBZ) fast frame buffer boards.
Stress mode	Yes	Stress mode exercises the frame buffer as much as possible. The Random test generator, constructed as part of the verification effort, is used. Starting from a known seed, random primitives with random attributes are generated. The primitives are checked to ensure that they were rendered in the same way as on a known good system. The test is repeated ten times, with each random picture overlaying the previous one. Stress mode is not available on FFB2+ boards.

`ffbtest` Command-Line Syntax

`/opt/SUNWvts/bin/ffbtest` *standard_arguments* **`-o dev=`***device_name*, **`S=`***subtest_number*,**`F=`***#_of_subtest_loops*,**`B=`***#_of_test_loops*,**`P=`***test_pattern*

TABLE 25-3 `ffbtest` Command-Line Syntax

Argument	Description
dev=*device_name*	*device_name* is the relative path name of the device being tested with respect to `/dev/fbs`; The default is `ffb0`.
S=*subtest_number*	*subtest_number* is the test number of the subtest to be run. Select from the subtests below. You can run multiple subtests by adding the subtest numbers. For example, `n=0x3` runs both test 1 and test 2; `n=0x180` runs both test 0x080 and test 0x0100. Note that you do not need the leading zeros. • `n=0x00001 3DRAM` • `n=0x00002 3DRAM Logic` • `n=0x00004 RAMDAC` • `n=0x00008 Rendering Pipeline` • `n=0x00010 FastFill/Vertical Scroll` • `n=0x00020 Pixel Processor` • `n=0x00040 Picking` • `n=0x00080 Arbitration` • `n=0x00100 Stereo` More than one test can be selected by ORing subtest numbers. For example: `n = 0x00009` selects 3DRAM and Rendering Pipeline tests. A hex number must be preceded by `0x`, decimal numbers are also acceptable.
F=*#_of_subtest_loops*	Specifies the number of times to repeat each subtest. The default is 1.
B=*#_of_test_loops*	Specifies the number of times to repeat a test loop before passing. The default is 1.
P=*test_pattern*	Specifies the test pattern number. The default is `r`, for random patterns. You may also choose `0` for `0x0000000`, `3` for `0x3333333`, `5` for `0x5555555`, or `9` for `0x9999999`.

Note – 64-bit tests are located in the `sparcv9` subdirectory: `/opt/SUNWvts/bin/sparcv9/`*testname*. 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 “32-Bit and 64-Bit Tests” on page 3.

Note – Errors returned by `ffbtest` are nonspecific. It is not possible to determine which component caused a failure. In all error conditions, the field replaceable unit (FRU) is the entire FFB.

CHAPTER 26

Floating Point Unit Test (`fputest`)

`fputest` checks the floating point unit on machines with SPARC architecture.

`fputest` Subtests

Instruction tests:

- 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:

- Linpack test
- Cparanoia test
- Kcsqrt test
- Kcdiv test
- Clorenz test
- Cvector test

`fputest` Options

To reach the 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 may not include the device appropriate to this test. Refer to the *SunVTS User's Guide* for more details.

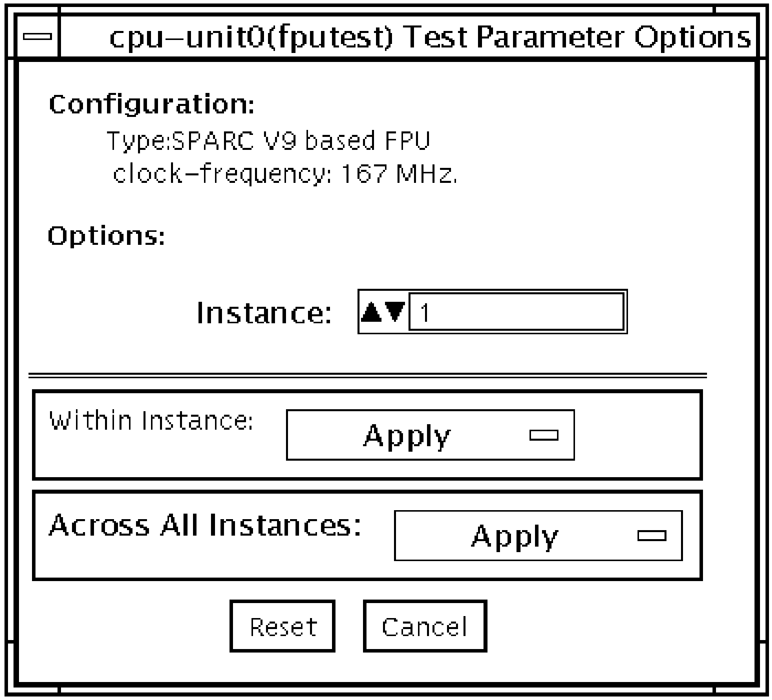


FIGURE 26-1 `fputest` Test Parameter Options Dialog Box

`fputest` Test Modes

TABLE 26-1 `fputest` Test Modes

Test Mode	Supported?	Description
Connection	Yes	In this mode, `fputest` includes all the instruction tests.
Functional (Offline)	Yes	Runs all the instruction tests and all the benchmark tests.
Stress mode	Yes	This mode runs several `fpu` benchmark tests.

`fputest` Command-Line Syntax

`/opt/SUNWvts/bin/fputest` *standard_arguments*

Note – 64-bit tests are located in the `sparcv9` subdirectory: `/opt/SUNWvts/bin/sparcv9/`*testname*. 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 "32-Bit and 64-Bit Tests" on page 3.

CHAPTER 27

IEEE 1394 Camera Test (fwcamtest)

fwcamtest tests the parameters and display functions of an IEEE 1394 digital camera, such as vid mode, frame rate, and frames received. This test currently supports the LG PC-10 camera.

Note – Do not run the fwcamtest and vmemtest at the same time on any Sun Blade™ system. This may cause the tests to fail.

Note – Do not start Sunforum™ (or any application that uses the dcam0 device) while fwcamtest is running. This causes the test to fail.

---

fwcamtest Test Requirements

Start a Window Environment

The system that runs fwcamtest must already be running a window environment, such as CDE or Open Windows. If the system has no window environment, or is only displaying the login window, fwcamtest will neither pass nor fail.

Note – Your window system must be operating in 24-bit depth to run the display test. Instructions for changing this setting are below.

If you are working in CDE, you can change your system to 24-bit depth by editing the file `/usr/dt/config/Xservers` or `/etc/dt/config/Xservers`. The file `/etc/dt/config/Xservers` overrides the file `/usr/dt/config/Xservers`. Edit the appropriate file to include the following line:

```
:0   Local local_uid@console root /usr/openwin/bin/Xsun :0
-nobanner -dev /dev/fbs/ffb0 defdepth 24 defclass TrueColor
```

If you are working in Open Windows, start Open Windows by typing the following at the shell prompt:

```
% openwin -dev /dev/fbs/ffb0 defdepth 24
```

Testing Through a Remote Connection

While running `fwcamtest` through a remote connection (such as a telnet session), if the `DISPLAY` variable is not set properly, it will cause numerous warning messages to display. These messages are logged and can fill up the log files. To avoid this, set your `DISPLAY` variable for the local host, and perform `xhost` *remote_host* on the local host before you start SunVTS and run `fwcamtest`.

`fwcamtest` Subtests

`fwcamtest` has three subtests:

- Parameter Test—Tests the digital camera parameters such as vid mode and brightness.
- Framereceive Test—Initializes the vid mode, framerate and ring buffer capacity parameters, then checks for the frame received.
- Display Test—Displays the captured frames. This display test sets up the 1394 bus for asynchronous transfer mode. The display test will only display the frames on the host running the test; it cannot display on a remote host.

fwcamtest Options

To reach the 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 may not include the device appropriate to this test. Refer to the *SunVTS User's Guide* for more details.

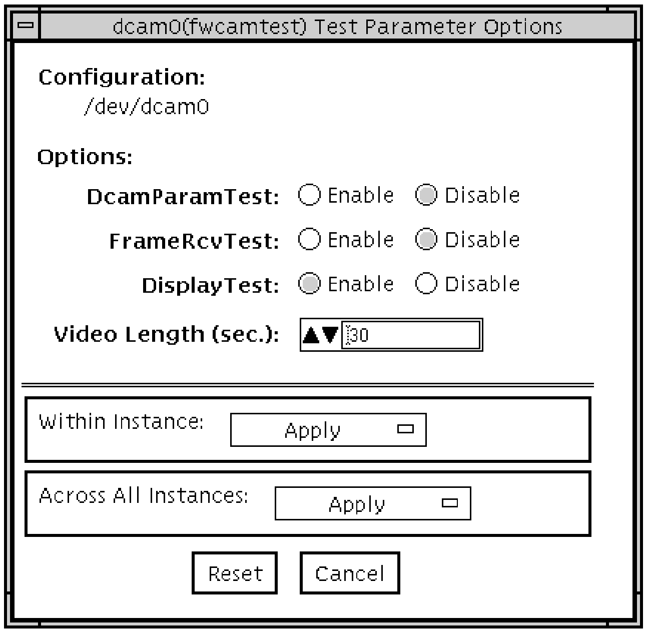


FIGURE 27-1 `fwcamtest` Test Parameter Options Dialog Box

TABLE 27-1 `fwcamtest` Options

`fwcamtest` **Options**	**Description**
DcamParamTest	Enables or disables the Parameter subtest.
FrameRevTest	Enables or disables the Frame Receive subtest.
DisplayTest	Enables or disables the Display subtest.
Video Length	Determines, in seconds, how long the screen display lasts.

`fwcamtest` Test Modes

TABLE 27-2 `fwcamtest` Test Modes

Test Mode	Supported?	Description
Connection	No	Not supported.
Functional (Offline)	Yes	Runs the full set of tests.

`fwcamtest` Command-Line Syntax

`/opt/SUNWvts/bin/fwcamtest` *standard_arguments*
`-o dev=`*dcam0***`,dcamparam=`***E(nable)* | *D(isable)***`,framercv=`***E* | *D***`,display=`***E* | *D***`,`**
`T=`*seconds*

TABLE 27-3 `fwcamtest` Command-Line Syntax

Argument	Description
`dev=`*dcam0*	Specifies the device name for this test.
`dcamparam=`*E(nable)* | *D(isable)*	Enables or disables the Parameter subtest.
`framercv=`*E* | *D*	Enables or disables the Frame Receive subtest.
`display=`*E* | *D*	Enables or disables the Display subtest.
`T=`*seconds*	Specifies the time period of the display test in seconds.

Note – 64-bit tests are located in the `sparcv9` subdirectory: `/opt/SUNWvts/bin/sparcv9/`*testname*. If the test is not present in this directory, then it may only be available as a 32-bit test. For more information refer to “32-Bit and 64-Bit Tests” on page 3.

CHAPTER **28**

PGX32 Frame Buffer Test (`gfxtest`)

The `gfxtest` tests the PGX32 frame buffer card by performing video memory, RAMDAC, and acceleration subtests as described in TABLE 28-1.

TABLE 28-1 `gfxtest` Subtests

Subtest	Description
Video Memory	Thoroughly tests the 8-Mbytes of video frame buffer memory using random and sequential 8-bit and 32-bit accesses. One pass of this subtest takes a while to complete.
RAMDAC	Tests the internal circuitry of the video output. The RAMDAC CLUT (color Lookup table) is tested using simple/read-write patterns to determine if there are any bad bits in the CLUT. This test only checks the functionality of the digital portion of the RAMDAC. This subtest does not ensure that the analog video signals are produced properly. This subtest takes less then one second to complete.
Acceleration	Thoroughly tests the hardware graphics acceleration of the PGX32 card. It re-initializes the engine, tests primitives such as lines, rectangles, bitblts, tiled and stippled rectangles with on-screen and off-screen memory. The Acceleration test also tests the ALU. This test takes less than one second to complete.

`gfxtest` Test Requirements

Caution – DO NOT run any other application or screen saver program that uses the PGX32 frame buffer card. These programs cause SunVTS to return incorrect errors.

Disable all screen savers before testing any graphics device. Type **xset s off** at a UNIX prompt to disable the Solaris screen saver.

Due to the nature of graphics tests, reading from or writing to the frame buffer during graphics tests will disturb user operation.

Do not run gfxtest from the SunVTS TTY mode when you are at the console.

For full instructions on testing frame buffers, see "Testing Frame Buffers" on page 9.

Note – DO NOT run Open Windows across multiple monitors.

Note – To start SunVTS with vtsui, but without vtsk, you may need to add the host name to xhost as: **xhost +** *hostname*.

---

gfxtest Options

To reach the 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 may not include the device appropriate to this test. Refer to the *SunVTS User's Guide* for more details.

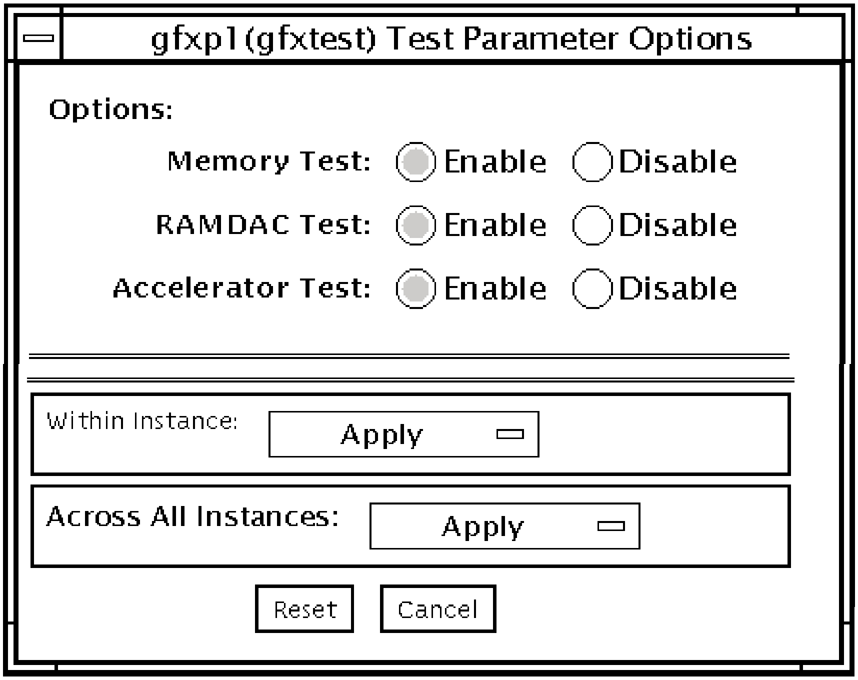


FIGURE 28-1 `gfxtest` Test Parameter Options Dialog Box

TABLE 28-2 `gfxtest` Options

Options	Description
Memory Test	Enables or disables the Video Memory subtest. The default is enabled.
RAMDAC Test	Enables or disables the RAMDAC subtest The default is enabled.
Acceleration Test	Enables or disables the Acceleration subtest. The default is enabled.

`gfxtest` Test Modes

TABLE 28-3 `gfxtest` Test Modes

Test Mode	Supported?	Description
Connection	No	Not supported.
Functional (Offline)	Yes	Runs the full set of tests.

`gfxtest` Command-Line Syntax

`/opt/SUNWvts/bin/gfxtest` *standard_arguments* **`-o`**
`dev=`*device_name***`,mem=`***no_of_passes***`,ramdac=`***no_of_passes***`,accel=`***no_of_passes*

TABLE 28-4 `gfxtest` Command-Line Syntax

Argument	Description
`dev=`*device_name*	*device_name* is the relative path name of the device being tested with respect to `/dev/fbs`, for example, `/dev/fbs/gfxp0`.
`mem=`*no_of_passes*	Specifies the number of times to run the Video Memory subtest. The default is one time.
`ramdac=`*no_of_passes*	Specifies the number of times to run the RAMDAC subtest. The default is one time.
`accel=`*no_of_passes*	Specifies the number of times to run the Acceleration subtest. The default is one time.

Note – 64-bit tests are located in the `sparcv9` subdirectory: `/opt/SUNWvts/bin/sparcv9/`*testname*. 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 “32-Bit and 64-Bit Tests” on page 3.

Note – Errors returned by `gfxtest` are nonspecific. It is not possible to determine which component caused a failure. In all error conditions, the field replaceable unit (FRU) is the entire PGX32 frame buffer card.

CHAPTER **29**

I2C Bus Test (i2ctest)

`i2ctest` checks each I2C bus for the status of its devices. `i2ctest` then determines any device faults based on the information it collects, and displays a report. `i2ctest` also detects and reports hung I2C bus segments.

This test is scalable.

Note – Only the 64-bit version of this test is supported.

---

i2ctest Test Requirements

- You must have the `SUNWpiclh`, `SUNWpiclr`, `SUNWpiclu`, and `SUNWpiclx` picl packages installed correctly before running the test.
- Verify that the `picld` daemon is running by typing the following:

```
# ps -ef |grep picld
root 100077      1  0   Sep 11 ?       23:40 /usr/lib/picl/picld
```

If the daemon is not running, run the script to restart it by typing the following:

```
# /etc/init.d/picld start
```

i2ctest Options

To reach the 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 may not include the device appropriate to this test. Refer to the *SunVTS User's Guide* for more details.

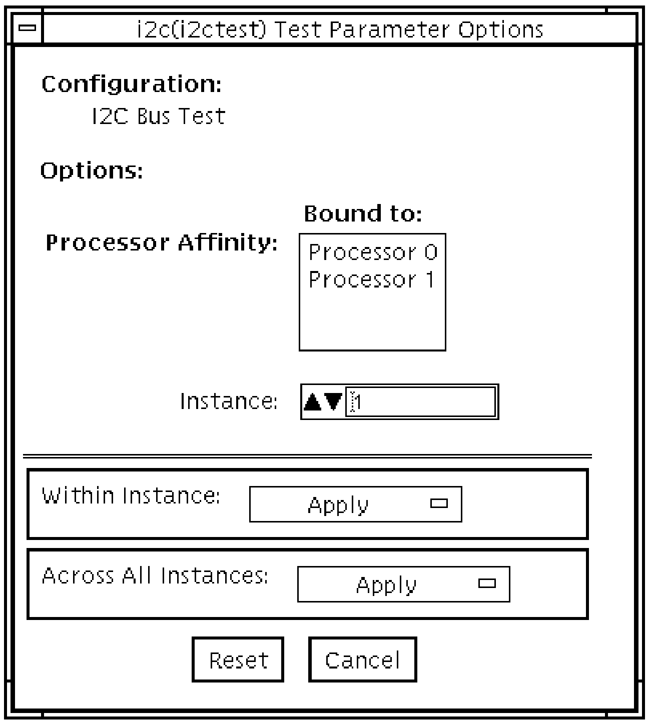


FIGURE 29-1 `i2ctest` Test Parameter Options Dialog Box

`i2ctest` Test Modes

TABLE 29-1 `i2ctest` Test Modes

Test Mode	Supported?	Description
Connection	Yes	Runs the full test.
Functional (Offline)	Yes	Runs the full test.

`i2ctest` Command-Line Syntax

/opt/SUNWvts/bin/sparcv9/i2ctest *standard_arguments*
-o dev=*raw_device_name*

TABLE 29-2 `i2ctest` Command-Line Syntax

Argument	Description
dev=*raw_device_name*	Specifies the name of the raw device to test.

CHAPTER 30

Expert3D Frame Buffer Test (ifbtest)

ifbtest verifies the functionality of the Expert3D frame buffer.

ifbtest can detect and adapt to many video modes of the Expert3D frame buffer. All tests can run at a resolution of 1024x768 or higher.

You can interrupt ifbtest using Control-C. Turn off all other keyboard input if OPEN LOOK is running on the unit being tested.

Test accuracy is checked using direct image comparison against compressed images. Failed pixel locations are printed as error messages.

Caution – Do not run any other application or screen saver program that uses the Expert3D accelerator port while running ifbtest. This combination causes SunVTS to return incorrect errors.

ifbtest Test Requirements

Disable all screen savers before testing any graphics device. To disable the Solaris screen saver, type the following at a UNIX prompt:

```
# xset s off
```

The display resolution must be 1024x768 or higher (the standard resolution). To change resolution, go to a UNIX prompt and type:

```
# fbconfig -res 1280x1024x76
```

Note – Do not run Open Windows across multiple monitors while running `ifbtest`. The test will return errors.

Preparation for `ifbtest`

You should complete a few steps in advance to ensure that `ifbtest` runs as smoothly as possible.

If you are running `ifbtest` in a window system (such as CDE or OPEN LOOK):

- Turn off Power Management if it is enabled.
- Make sure that no other program is running that might modify the screen during the test.
- Make sure you have permission to lock the X server. `ifbtest` is designed to lock the X server during testing to prevent screen changes.
- The CDE login window should not be displayed during testing.
- Check that the window system is only running on one Expert3D frame buffer.

If you are not running `ifbtest` in a window system:

- Turn off Power Management if it is enabled.
- Make sure that no other program is running that might modify the screen during the test.
- Make sure the Expert3D frame buffer being tested is not the console device. Console messages may modify the screen.

`ifbtest` Options

To reach the 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 may not include the device appropriate to this test. Refer to the *SunVTS User's Guide* for more details.

By default, all `ifbtest` options are enabled.

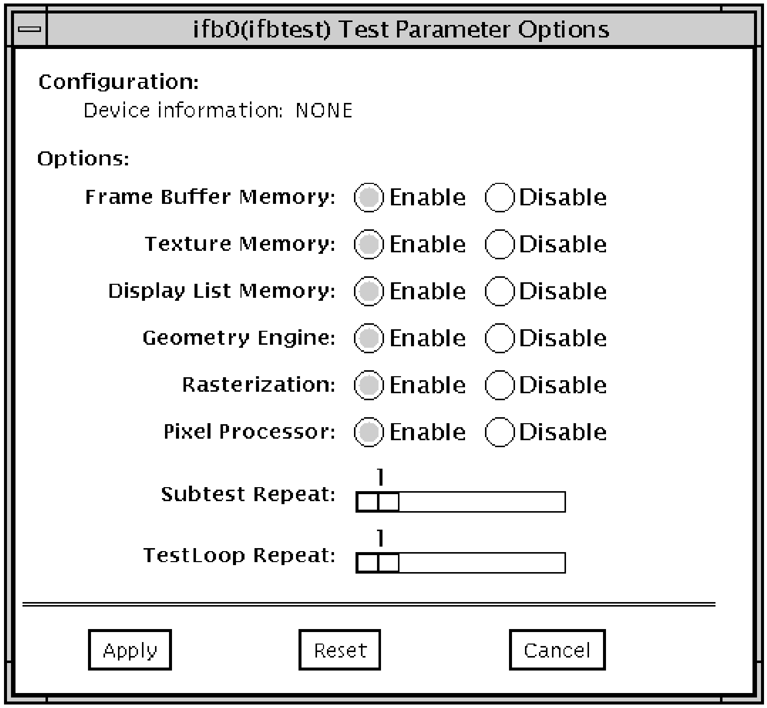


FIGURE 30-1 `ifbtest` Test Parameter Options Dialog Box

TABLE 30-1 `ifbtest` Options

`ifbtest` **Options**	**Description**
Frame Buffer Memory test	Thoroughly tests the Expert3D video memory by using read and write requests. Tests for shorts or failed connections on the data bus by writing the following values to every address: • `0xFFFFFFFF` • `0xFFFF0000` • `0x0000FFFF` • `0xFF00FF00` • `0x00FF00FF` • `0xF0F0F0F0` • `0x0F0F0F0F` • `0xCCCCCCCC` • `0x33333333` • `0xAAAAAAAA` • `0x55555555` Tests for shorts or failed connections on the address bus by writing the offset of each memory location to each location and reading them back. This may also catch speed-related problems due to the volume of read/writes. Errors in the test are reported as an error in a particular address, not attributed to a specific chip. To help distinguish bit-related errors, the errors are summarized to list which bits had at least one error in the test. This test shows on the screen as random pixels.
Texture Memory test	This test is identical in process to the frame buffer memory test (above). Since this test produces no visible effect, rectangles are drawn in rows across the screen to show progress.
Display List Memory test	This test is identical in process to the frame buffer memory and texture memory tests (above), and is applied to direct burst memory. This test takes little time and no progress is displayed.
Geometry Engine test	Loads diagnostic microcode into the geometry engine and confirms that the processor operates correctly. This is a pass/fail test. This test takes little time and no progress is displayed.

TABLE 30-1 `ifbtest` Options

`ifbtest` **Options**	**Description**
Rasterization test	Renders many primitives with minimal fragment processing, to test the rasterization of the primitives. The primitives used are: • Dots • Anti-aliased dots • Lines using all for line-drawing primitives • Anti-aliased lines using all for line-drawing primitives • Triangles, Quads, and Polygons in point, line, and fill modes • Rectangles This tests for the following rasterization attributes: • pixel coverage • constant value registers for color, Z, and stencil • interpolation of color, Z, and texture coordinates along lines and spans in polygons • texture map sampling Resulting images are compared against stored images. Errors indicate which operation type and value was being tested, and the coordinate of the failed pixel.

TABLE 30-1 `ifbtest` Options

`ifbtest` **Options**	**Description**
Pixel Processor test	Tries the various pixel processing operators using a variety of fragment values. This tests the following fragment processing operations: • Depth Buffering • Blending • Alpha Test • Color Test • Color Clamp • Logic Operations • Color Matrix and Bias • Color Table • Control Planes • Fast Clear • Stencil • Scissor Clipping • Desktop Clipping • Mask Clipping • Write Masks • Window Origin • Fog • Pixel Texture • Accumulation Buffer • Pixel Buffers Resulting images are compared against stored images. Errors indicate which operation type and value was being tested and the coordinate of the failed pixel.

`ifbtest` Test Modes

Due to the nature of graphic tests, reading data from, or writing data to the frame buffer during graphic tests will disturb user operation. For this reason, `ifbtest` is only available in Offline Functional test mode.

TABLE 30-2 `ifbtest` Test Modes

Test Mode	Supported?	Description
Connection	No	Not supported.
Functional (Offline)	Yes	Runs the full set of tests.

`ifbtest` Command-Line Syntax

`/opt/SUNWvts/bin/ifbtest` *standard_arguments* **`-o dev=`***device_name***`,`** **`fbmem=`***E(nable)/D(isable)***`,texmem=`***E/D***`,dlmem=`***E/D***`,geomeng=`***E/D***`,`** **`rasterization=`***E/D***`,pixelproc=`***E/D***`,subtest_repeat=`***number***`,`** **`test_repeat=`***number*

TABLE 30-3 `ifbtest` Command-Line Syntax

Argument	Description
`dev=`*device_name*	*device_name* is the relative path name of the device being tested with respect to `/dev/fbs`. There is no default.
`fbmem=`*E/D*	Enables or disables the frame buffer memory test.
`texmem=`*E/D*	Enables or disables the texture memory test.
`dlmem=`*E/D*	Enables or disables the display list memory test.
`geomeng=`*E/D*	Enables or disables the geometry engine test.
`rasterization=`*E/D*	Enables or disables the rasterization test.
`pixelproc=`*E/D*	Enables or disables the pixel processing test.
`subtest_repeat=`*number*	Defines the number of times to repeat each subtest. The default is 1.
`test_repeat=`*number*	Defines the number of times to repeat a test loop before passing. The default is 1.

Note – 64-bit tests are located in the `sparcv9` subdirectory: `/opt/SUNWvts/bin/sparcv9/`*testname*. If the test is not present in this directory, then it may only be available as a 32-bit test. For more information refer to “32-Bit and 64-Bit Tests” on page 3.

CHAPTER **31**

Intelligent Fibre Channel Processor Test (`ifptest`)

`ifptest` tests the functionality of the PCI FC-AL card when there are no devices attached to the loop. The driver checks for devices on the fibre loop. If devices are detected the driver blocks any diagnostic commands.

Note – When devices are attached to the loop, do not run `ifptest`. Instead, run `disktest` tests on the individual devices. This will test the whole subsystem including the FC-AL controller.

`ifptest` uses the "mailbox" interface to the card. This interface allows certain firmware operations to be performed that normally would not be available to the application layer.

`ifptest` Subtests

Four subtests are run in online and functional modes:

- Mailbox Loopback test

Loads a series of registers into the input mailboxes on the card and then reads the output mailboxes and compares results. This verifies that the system side of the card is operating correctly, and that the internal data paths are okay.

- Firmware revision check

Reads the firmware revision from the firmware and compares it against a revision loaded by the driver.

- Checksum firmware test

Runs an internal checksum test on the installed firmware. This verifies that the RISC RAM on the card is fully functional and that the installed firmware is still intact. This test also serves as a quick RAM check of the RISC RAM.

- Dump revision levels

Extracts the hardware and firmware revision levels of different submodules on the card.

`ifptest` Options

To reach the 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 may not include the device appropriate to this test. Refer to the *SunVTS User's Guide* for more details.

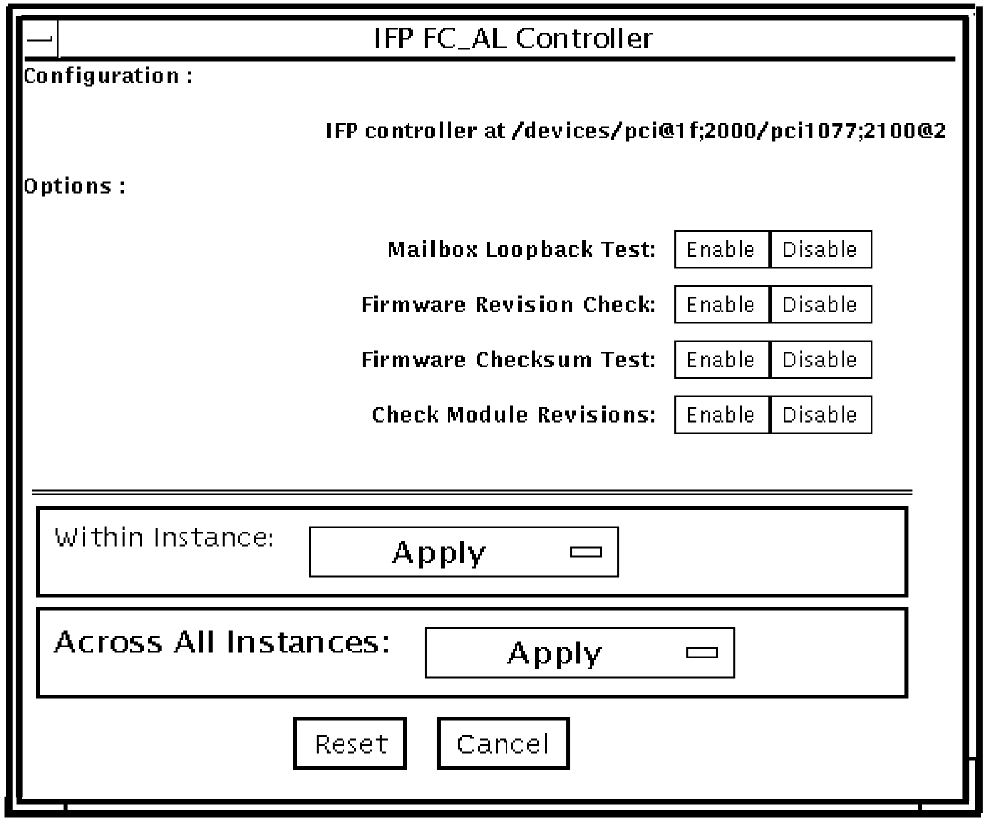


FIGURE 31-1 `ifptest` Test Parameter Options Dialog Box

TABLE 31-1 `ifptest` Options

`ifptest` **Options**	**Description**
Mailbox Loopback test	Enables or disables the mailbox loopback command. This test writes data patterns into the mailboxes and then reads them back from the output mailboxes and verifies the data is correct.

TABLE 31-1 `ifptest` Options

`ifptest` **Options**	**Description**
Firmware revision check	Enables or disables the firmware revision check command. This test extracts the firmware revision from the RISC firmware code and verifies against expected values.
Firmware checksum test	Enables or disables the firmware checksum command. This command instructs the interface's RISC processor to calculate the checksum on the current microcode and then compare it against the checksum that was loaded in with the microcode.
Check Module Revisions	Enables or disables the firmware check module command. This command returns the revision level of several submodules on the interface card. Although this test is executed when enabled, the module revision levels are only printed out in VERBOSE mode.

`ifptest` Test Modes

TABLE 31-2 `ifptest` Test Modes

Test Mode	**Supported?**	**Description**
Connection	Yes	Performs only an open/close operation.
Functional (Offline)	Yes	Runs the full set of mailbox tests.

Note – Connection test mode will only open the controller to verify that the path is still viable.

`ifptest` Command-Line Syntax

`/opt/SUNWvts/bin/ifptest` *standard_arguments*
`-o dev=`*device name***`,mbox=`***Enable | Disable***`,fwrevcheck=`***Enable | Disable***`,`**
`checksum=`*Enable | Disable***`,modrevcheck=`***Enable | Disable*

TABLE 31-3 `ifptest` Command-Line Syntax

Argument	Description
`dev=`	The name of the device to test.
`mbox=`*Enable | Disable*	Enables or disables the mailbox loopback command. This test writes data patterns into the mailboxes and then reads them back from the output mailboxes and verifies the data is correct.
`fwrevcheck=` *Enable | Disable*	Enables or disables the firmware revision check command. This test extracts the firmware revision from the RISC firmware code and verifies against expected values.
`checksum=` *Enable | Disable*	Enables or disables the firmware checksum command. This command instructs the interface's RISC processor to calculate the checksum on the current microcode and then compare it against the checksum that was loaded in with the microcode.
`modrevcheck=` *Enable | Disable*	Enables or disables the firmware checksum command. This command returns the revision level of several sub-modules on the interface card. Although this test is executed when enabled, the module revision levels are only printed out in VERBOSE mode.

Note – 64-bit tests are located in the `sparcv9` subdirectory: `/opt/SUNWvts/bin/sparcv9/`*testname*. 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 “32-Bit and 64-Bit Tests” on page 3.

CHAPTER 32

Dual Basic Rate ISDN (DBRI) Chip (`isdntest`)

`isdntest` verifies the functionality of the ISDN portion of the Dual Basic Rate ISDN (DBRI) chip.

`isdntest` Subtests

`isdntest` is actually a set of several subtests. Three main channels exist within an ISDN: D, B1 and B2. Each channel runs as an independent thread. In each of the following subtests (unless otherwise indicated), the settings are as follows:

TABLE 32-1 `isdntest` Channel Settings

Channel	Data Mode	Packet Size	Packet Count
D	Basic Rate HDLC	256 bytes	10 packets
B1	56-kbps HDLC	1024 bytes	10 packets
B2	64-kbps HDLC	1024 bytes	10 packets

TABLE 32-2 `isdntest` Subtests

`isdntest` **Subtests**	**Description**
Local Loopback test	The local loopback test checks the initial activation state of the Network Termination (NT) and Terminal Equipment (TE) interfaces to make sure they are deactivated. The test then activates each interface using the “force activation” capability of DBRI. Each interface is put into local loopback mode. Data residing in host memory is written to each interface, which loops the data back to itself. The data is then read back into host memory and verified. Each channel—D, B1, and B2—is tested (with the exception of the TE D channel, which cannot be tested in local loopback mode). The local loopback test runs internal to the DBRI chip and does not require an NT-to-TE external loopback connector.
Activation / Deactivation test	The Activation / Deactivation test runs through the Activation / Deactivation sequence for the NT and then the activation sequence for the TE. The T101 and T103 timers are set to five seconds. This subtest requires an NT-to-TE external loopback connector.
Remote Loopback test	The remote loopback capability is tested next. The TE interface is put into remote loopback mode, and the NT transmits data to the TE on all three channels, D, B1 and B2. The TE loops all data back to the NT and reads a copy of it. Data is then verified. Next, the whole process is repeated with the TE transmitting to the NT, which is placed in remote loopback mode. This subtest requires an NT-to-TE external loopback connector.
Read / Write test	Next, the Read / Write test is performed on all six ISDN channels: TE D, TE B1, TE B2, NT D, NT B1, and NT B2. The external loopback connector connects each channel on the TE interface to its corresponding channel on the NT. A unique data pattern is used for each path. Packets read are compared against packets written. The test is repeated with the B1 channels placed in 64-kbps HDLC data mode and the B2 channels in 56-kbps HDLC data mode. This subtest requires an NT-to-TE external loopback connector.
Packet Size test	The next subtest is the Packet Size test. A read / write test, similar to the previous one, is performed with a packet count of 100. Each packet transmitted and received is a unique size, computed randomly. This subtest requires an NT-to-TE external loopback connector.
Data Path test	The last subtest is the Data Path test. Using the ISDN_SET_CHANNEL `ioctl`, data is routed through a series of short pipe interconnects within DBRI. This subtest requires an NT-to-TE external loopback connector.

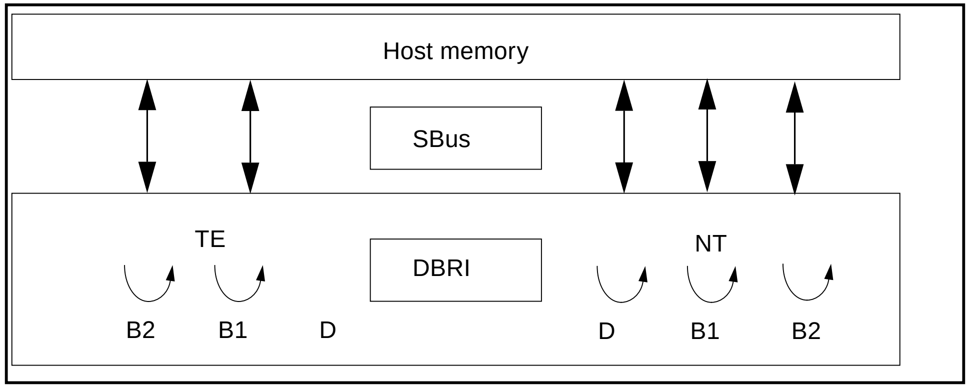


FIGURE 32-1 `isdntest` Local Loopback Subtest

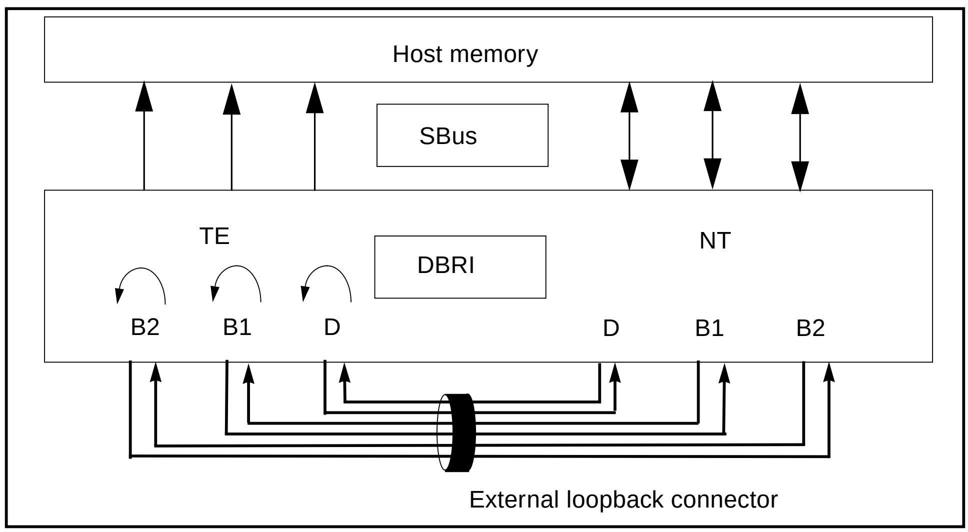


FIGURE 32-2 `isdntest` Remote Loopback Subtest

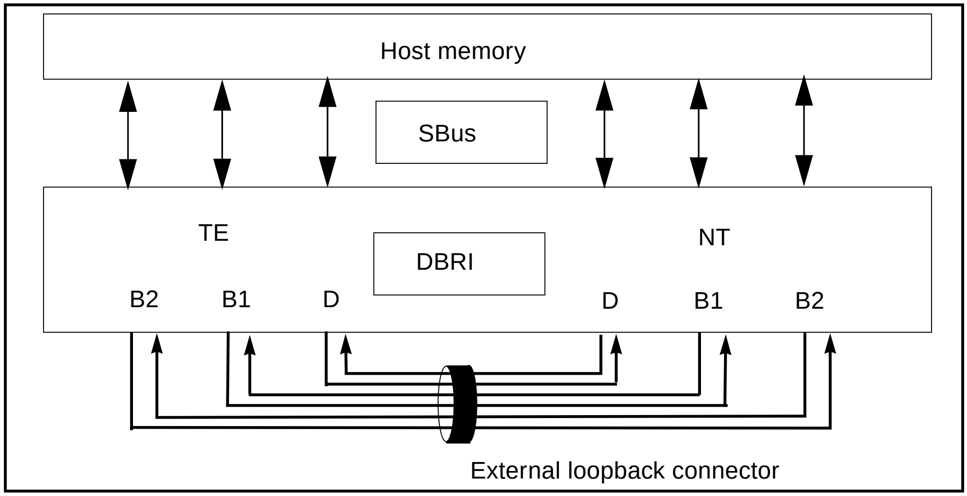


FIGURE 32-3 `isdntest` Read / Write Subtest

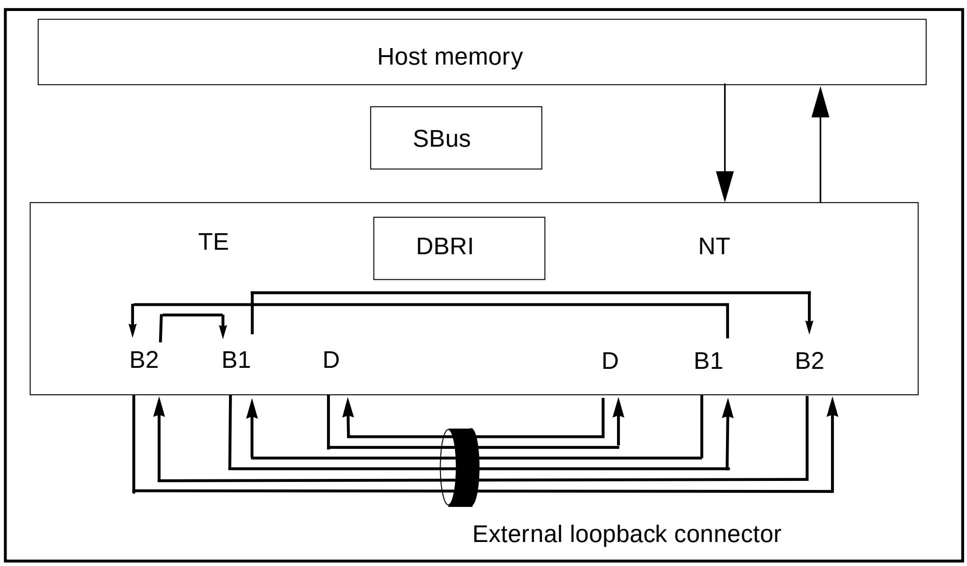


FIGURE 32-4 `isdntest` Data Path Subtest

isdntest Options

To reach the 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 may not include the device appropriate to this test. Refer to the *SunVTS User's Guide* for more details.

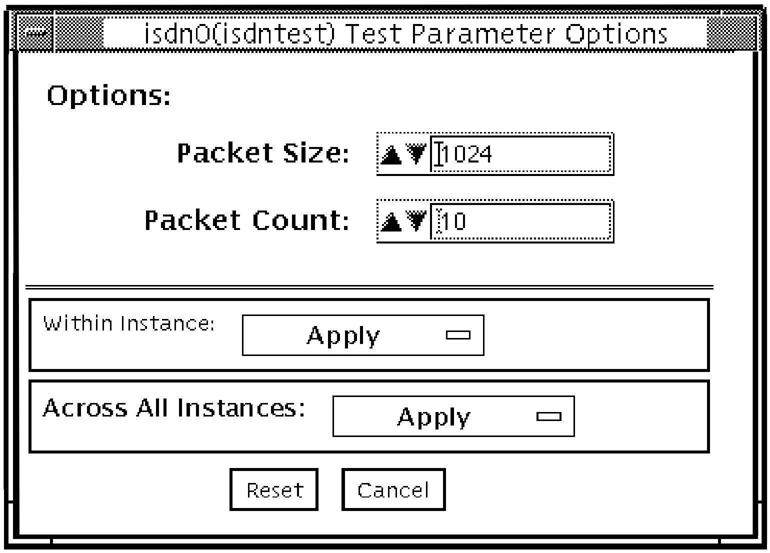


FIGURE 32-5 isdntest Test Parameter Options Dialog Box

TABLE 32-3 isdntest Options

isdntest **Options**	**Description**
Packet Size	The byte size of the B channel packets. The default size is 1024 bytes for the B channels and 256 for the D channels. The maximum packet size is 8186 bytes for the B channels, and the minimum packet size is 1 byte. The D channel packet size is set to 256, except during the packet size test, when it is set to random values between 1 and 256.
Packet Count	Number of packets to be transmitted and received for all channels. The default packet count is 10 packets. The maximum packet count is 100 packets.

`isdntest` Test Modes

TABLE 32-4 `isdntest` Test Modes

Test Mode	Supported?	Description
Connection	No	Not supported.
Functional (Offline)	Yes	Runs the full set of tests.

`isdntest` Command-Line Syntax

`/opt/SUNWvts/bin/isdntest` *standard_arguments* **`-o size=`***packet_size***`,`**
`count=`*packet_count*

TABLE 32-5 `isdntest` Command-Line Syntax

Argument	Description
`size=`*packet_size*	The byte size of the B channel packets. The B channel default size is `1024` bytes. The D channel default size is 256 bytes. The B channel maximum packet size is 8186 packets. The B channel minimum packet size is 1 packet. The D channel packet size is 256 (except during the packet size test, when it is set to random values between 1 and 256).
`count=`*packet_count*	Number of packets to be transmitted and received for all channels. The default count is 10 packets. The maximum packet count is 100 packets.

Note – 64-bit tests are located in the `sparcv9` subdirectory: `/opt/SUNWvts/bin/sparcv9/`*testname*. 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 “32-Bit and 64-Bit Tests” on page 3.

CHAPTER 33

Level 1 Data Cache Test (`l1dcachetest`)

`l1dcachetest` exercises the level1 D cache in the CPU module of a Sun system. The test writes, reads, and verifies access of multiple virtual addresses. The test determines addresses by the buffer size, which is chosen according to the size of the l1dcache. The test writes to an address bigger than the cache size, to cause thrashing (cache misses) in up to 4-way set associative data caches.

`l1dcachetest` Options

To reach the 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 may not include the device appropriate to this test. Refer to the *SunVTS User's Guide* for more details.

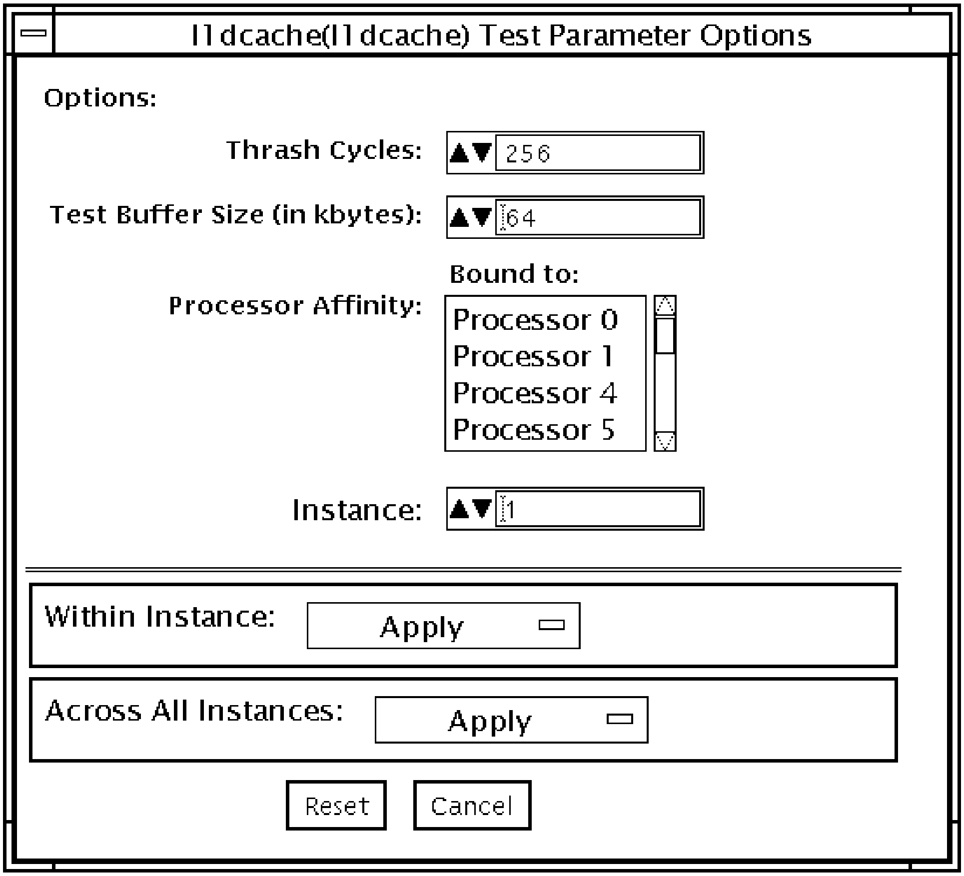


FIGURE 33-1 `l1dcachetest` Test Parameter Options Dialog Box

TABLE 33-1 `l1dcachetest` Options

Option	Description
Thrash Cycles	Specifies the number of thrashing cycles the test completes for the level1 cache on the system. Default value for Offline mode is 256.
Test Buffer Size	Sets the size of the buffer, in kbytes, that the test allocates for testing. Default value is 64.

`l1dcachetest` Test Modes

TABLE 33-2 `l1dcachetest` Test Modes

Test Mode	Supported?	Description
Connection	No	Not supported.
Functional (Offline)	Yes	Runs the full test.

`l1dcachetest` Command-Line Syntax

`/opt/SUNWvts/bin/l1dcache` *standard_arguments*
`-o count=`*number***`,buffer=`***number***`,dev=`***l1dcache*

TABLE 33-3 `l1dcachetest` Command-Line Syntax

Argument	Description
`count=`*number*	Specifies the number of thrashing cycles the test completes for the level1 cache on the system. Default value for Offline mode is 256.
`buffer=`*number*	Sets the size of the buffer, in kbytes, that the test allocates for testing. Default value is 64.
`dev=`*l1dcache*	Specifies the name of the device.

CHAPTER **34**

Level 2 Cache Test (`l2dcachetest`)

l2dcachetest exercises the level2 external cache in the CPU module of a Sun machine. The test writes, reads, and verifies access of multiple virtual addresses.The multiple virtual addresses are chosen to cause thrashing (cache misses) in direct-mapped external caches.

`l2dcachetest` Options

To reach the 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 may not include the device appropriate to this test. Refer to the *SunVTS User's Guide* for more details.

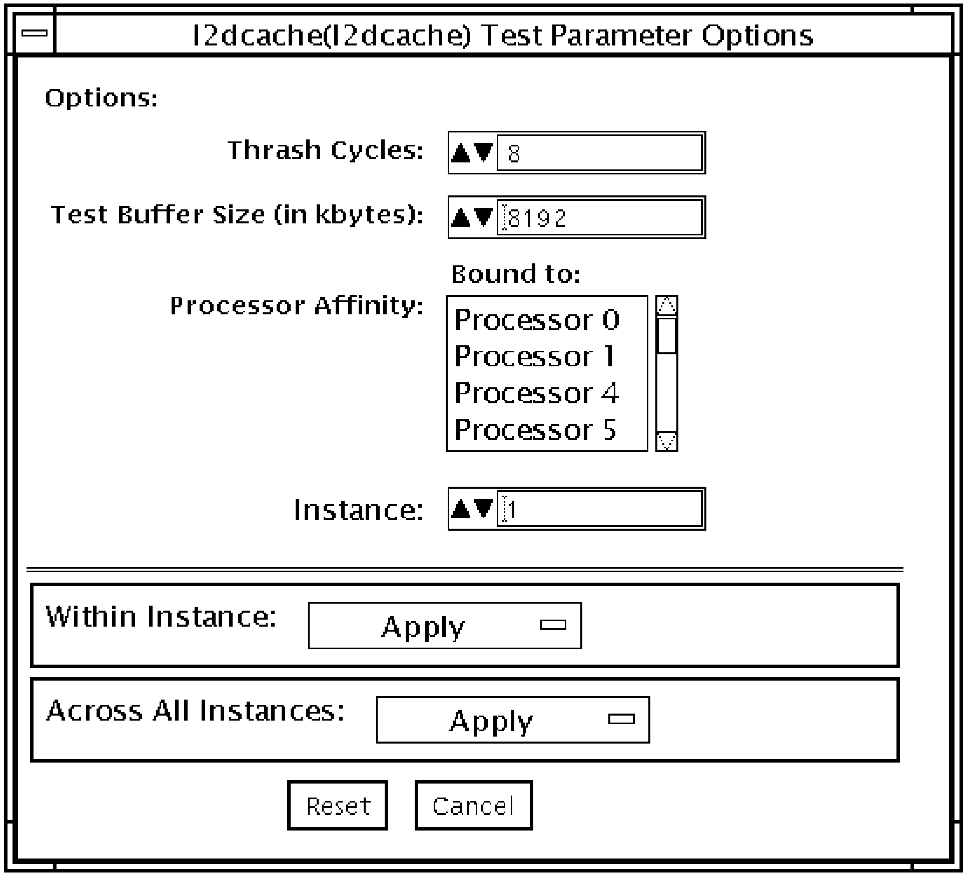


FIGURE 34-1 `l2dcachetest` Test Parameter Options Dialog Box

TABLE 34-1 `l2dcachetest` Options

Option	Description
Thrash Cycles	Specifies the number of thrashing cycles the test completes for the level2 cache on the system. Default value for Offline mode is 8.
Test Buffer Size	Sets the size of the buffer, in kbytes, that the test allocates for testing. Default value is 8192.

l2dcachetest Test Modes

TABLE 34-2 l2dcachetest Test Modes

Test Mode	Supported?	Description
Connection	No	Not supported.
Functional (Offline)	Yes	Runs the full test.

l2dcachetest Command-Line Syntax

/opt/SUNWvts/bin/l2dcache *standard_arguments*
-o count=*number***,buffer=***number***,dev=***l2dcache*

TABLE 34-3 l2dcachetest Command-Line Syntax

Argument	Description
count=*number*	Specifies the number of thrashing cycles the test completes for the level2 cache on the system. Default value for Offline mode is 8.
buffer=*number*	Sets the size of the buffer, in kbytes, that the test allocates for testing. Default value is 8192.
dev=*l2dcache*	Specifies the name of the device.

CHAPTER 35

LOMlite Alarm Test (`lomlitetest`)

`lomlitetest` tests the functionality of LOMlite and LOMlite 2 system monitoring, alarms, and lights-out management (LOM) processors currently used in Netra™ T platforms. This test also tests the legacy TSalarms alarm card used in some Netra t 11xx systems. `lomlitetest` exercises the hardware and device drivers for the LOMlite, LOMlite 2, or TSalarms device, and tests the system and environmental monitoring functions of the device.

This test is not scalable.

Note – During offline functional testing, messages from the LOM processor are seen on the system console. This is normal and does not indicate a fault.

`lomlitetest` Requirements

- The LOMlite or TSalarms device driver must be installed
- In the Netra t 11xx or Netra t 14xx platforms, the LOMlite or TSalarms plug-in card must be installed

`lomlitetest` Subtests

TABLE 35-1 `lomlitetest` Subtests

Subtest	Description
Connection subtest	Run for both LOMlite and TSalarms. Attempts to open the device nodes appropriate to the device being tested.
Passive Read subtest	Attempts to read the available data from the device being tested. Device data is read as follows: All devices: Alarm state LOMlite and LOMlite 2: • Fault LED state • Power supply state • Fans state • EEPROM event log LOMlite 2 only: • Power supply voltages • Enclosure and CPU temperatures
Active Alarms subtest	Reads, inverts, and re-reads each alarm state to ensure change took effect. Resets alarms to original state and checks that the state is correct. In the case of LOMlite and LOMlite 2 devices, reads the EEPROM event log and ensures that the expected events are recorded.
Active Fault LED subtest	Reads, inverts, rereads and resets the state of the fault indicator LED. Checks that the state changes are recorded in the EEPROM event log.

`lomlitetest` Options

This test has no programmable options unless it is run on a multi-processor system. On single-processor systems, the appropriate subtests are automatically selected depending upon the test mode and the type of device detected or specified on the command line.

FIGURE 35-1 shows the options menu for a multi-processor system. To reach the 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 may not include the device appropriate to this test. Refer to the *SunVTS User's Guide* for more details.

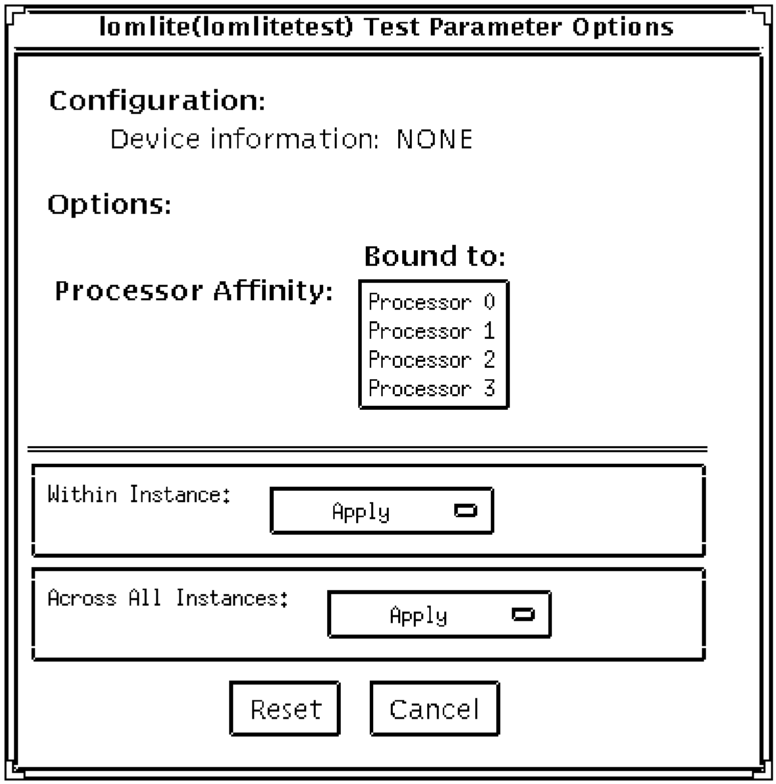


FIGURE 35-1 `lomlitetest` Test Parameter Options Dialog Box, Multi-processor System

`lomlitetest` Test Modes

TABLE 35-2 `lomlitetest` Test Modes

Test Mode	Supported?	Description
Connection	Yes	Runs the Connection subtest.
Functional (Offline)	Yes	Runs all subtests.

`lomlitetest` Command-Line Syntax

`/opt/SUNWvts/bin/lomlitetest` *standard_arguments*
`-o dev=`*lomlite2 | lomlite | tsalarms*

TABLE 35-3 `lomlitetest` Command-Line Syntax

Argument	Description
`dev=`*lomlite2 | lomlite | tsalarms*	Selects the type of device driver to test.

CHAPTER **36**

M64 Video Board Test (`m64test`)

`m64test` tests the PCI-based M64 video board by performing the following subtests:

- Video Memory test
- RAMDAC test
- Accelerator Port test

Caution – *Do not* run any other application or screen saver program that uses the M64 video board while running `m64test`. Do not run Power Management™ software. These programs cause SunVTS to return incorrect errors.

Note – Disable all screen savers before testing any graphics device. Type **`xset s off`** at a UNIX prompt to disable the Solaris screen saver. Disable Power Management software if it is running.

Note – *Do not* run Open Windows across multiple monitors.

Note – To start SunVTS with `vtsui`, but without `vtsk`, you must add the host name to `xhost` as: **`xhost +`** *hostname.*

`m64test` Options

To reach the 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 may not include the device appropriate to this test. Refer to the *SunVTS User's Guide* for more details.

By default, all options are enabled except frame buffer locking.

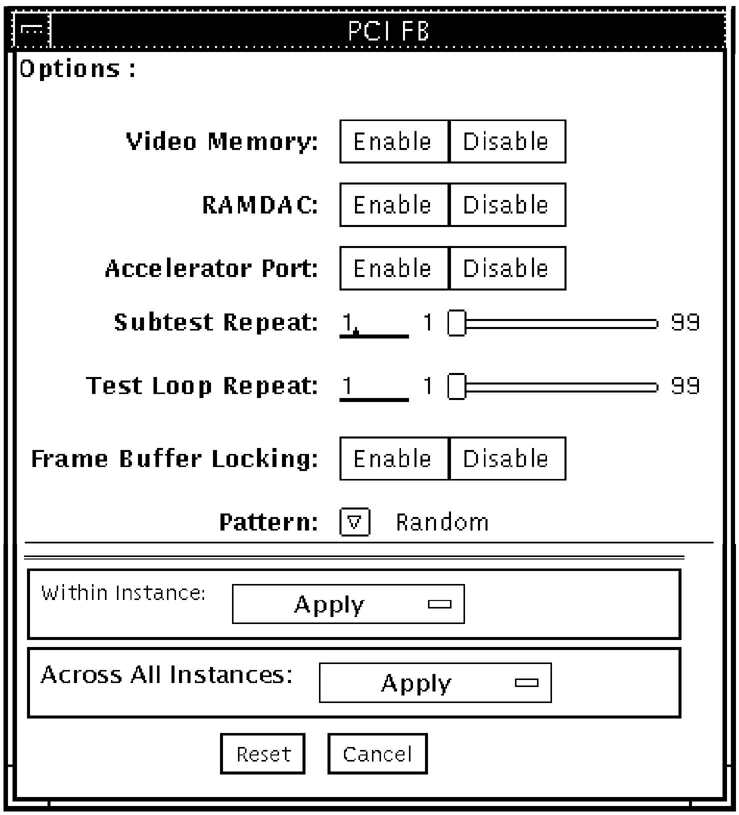


FIGURE 36-1 `m64test` Test Parameter Options Dialog Box

TABLE 36-1 `m64test` Options

`m64test` **Options**	**Description**
Video Memory test	Thoroughly tests the on-screen video memory (the memory part that is mapped on to the monitor) of the M64 video board in 8-bit, 16-bit, 32-bit, 64-bit, and 64 byte (block) modes. Entire on-screen video memory is tested by testing 512 bit blocks at a time (8x8 pixel block). Each block is tested in two passes. Each pass consists of a data write and read. In the first pass user specified data or random data is used and in the second pass one's complement of the data used in the first pass is used so that each on-screen video memory location (bit) is tested with a zero (electrical low state) and one (electrical high state).
RAMDAC test	Tests the RAMDAC in three phases. In the first phase the RAMDAC CLUT (Color LookUp Table) is tested using simple write/read patterns to determine if there are any bad bits in CLUT. The data patterns used are: • Random data • Complement of the random data (used as first data pattern) • The data pattern 0101 • The data pattern 10101 In the second phase, four different patterns are drawn on the screen. Each pattern stays on the screen for approximately three seconds. The four patterns are listed below. For each pattern the signature is captured and compared with the signature obtained for the same pattern on a known good board. This test verifies that all the different data paths within the RAMDAC are functioning properly. Patterns drawn on screen: • Red ramp with cursor at top-left corner of the screen • Blue ramp with cursor at top-right corner of the screen • Green ramp with cursor at bottom-left of the screen • Grey ramp with cursor at bottom-right of the screen In the last (third) phase of the RAMDAC test the Vertical Retrace Interrupt is tested for 300 interrupts.

TABLE 36-1 `m64test` Options

`m64test` Options	Description
Accelerator Port test	Tests all of the following: • Data paths (sources: fixed color, host data, blit, fixed pattern) • Arithmetic and logic unit (ALU) • Color comparator • Primitives (destinations: line, rectangle) • Mono to color expansion logic Primitives are drawn using a combination of different data paths (allowed), ALU functions, and color comparator functions. A checksum is generated for each data combination and is compared with the checksum generated for the same data combination on a known good board.
Frame Buffer Locking	This option is set to *disable* if the M64 is not the console device. When Sunvts GUI is brought up FB Locking is enabled by default if M64 is console device. If M64 is not console device, FB Locking is disabled by default.

`m64test` Test Modes

Due to the nature of graphics tests, reading from or writing to the frame buffer during graphics tests will disturb user operation. This test is only available in the Offline Functional test mode.

TABLE 36-2 `m64test` Test Modes

Test Mode	Supported?	Description
Connection	No	Not supported
Functional (Offline)	Yes	The `m64test` verifies the M64 video board.

`m64test` Command-Line Syntax

`/opt/SUNWvts/bin/m64test` *standard_arguments* **`-o dev=`***device_name***`,`**
`S=`*subtest_number*,**`F=`***#_of_subtest_loops*,**`B=`***#_of_test_loops***`,L=`***disable***`,P=`***test_pattern*

TABLE 36-3 `m64test` Command-Line Syntax

Argument	Description
dev=*device_name*	*device_name* is the relative path name of the device being tested with respect to /dev/fbs. The default is m640.
S=*subtest_number*	*subtest_number* is the test number of the subtest to be run. Select from the subtests below. You can run multiple subtests by adding the subtest numbers. For example, `n=0x00003` runs both test 00001 and test 00002; `n=0x00005` runs both test 0x00001 and test 0x00004. Note that you do not need the leading zeros. • `n—0x00001 VRAM` • `n—0x00002 RAMDAC` • `n—0x00004 Accelerator port test (Rendering Pipeline)` More than one test can be selected by ORing subtest numbers. For example: n = 0x00005 means VRAM and Rendering Pipeline tests. A hex number must be preceded by 0x, decimal numbers are also acceptable.
F=*#_of_subtest_loops*	Specifies the number of times to repeat each subtest. The default is 1.
B=*#_of_test_loops*	Specifies the number of times to repeat a test loop before passing; default is 1.
L=*disable*	Disables the frame buffer lock. Disable the lock when the m64 is not the console or when the server is not running on the m64 under test.
P=*test_pattern*	Specifies the test pattern number. The default is `r`, for random patterns. You may also choose `0 for 0x0000000`, `3 for 0x3333333`, `5 for 0x5555555`, or `9 for 0x9999999`.

Note – 64-bit tests are located in the `sparcv9` subdirectory: `/opt/SUNWvts/bin/sparcv9/`*testname*. 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 "32-Bit and 64-Bit Tests" on page 3.

Note – Errors returned by `m64test` are nonspecific. It is not possible to determine which component caused a failure. In all error conditions, the field replaceable unit (FRU) is the entire M64 video board.

CHAPTER **37**

Cache Consistency Test (`mpconstest`)

`mpconstest` verifies that cache coherency is maintained in a multi-processor environment by creating contention for one or more cache lines. This test has several subtests, each designed to create a different kind of contention for cache lines. Each subtest uses different methods to test the shared memory buffer, the stride size, and any intermediate stores or loads.

When `mpconstest` starts, it creates a shared memory buffer. It then determines the number of CPUs on the system. For each CPU, the test takes the following steps:

1. Forks a thread and binds it to the CPU.
2. Runs the selected subtest in the thread.
3. Assigns each CPU an ID number from 1 to *n*. The CPU assigned `ID 1` is considered the master.

The above steps are repeated for each subtest. Only one subtest can be selected at a time.

This test is not scalable.

`mpconstest` Test Requirements

This test requires that the tested system has at least two CPUs. Otherwise, the test will not appear as an option.

`mpconstest` only runs on machines that support the v8plus standard of SPARC CPU hardware architecture. If the v8plus instructions are not supported, `mpconstest` will not appear on the Test Selection GUI. To determine whether a machine supports the v8plus standard, go to a command prompt on that machine and type:

```
% isalist
sparcv9+vis sparcv9 sparcv8plus+vis sparcv8plus sparcv8
```

Note – This set of tests is very sensitive to activity on the machine and must be run exclusive of all other tests.

`mpconstest` Subtests

TABLE 37-1 `mpconstest` subtests

Tests	Description
cons1	Each CPU writes to successive locations with a stride size of byte, half word, or full word. This subtest creates contention for a single cache line. No other loads or stores are performed between successive writes to shared memory.
cons2	Each CPU reads from a location that is *cachesize* bytes away from the last written location. Every read causes the previously written line to be written back. The test runs until the CPU has accessed all lines in the cache.
cons3	Similar to cons1 except that only one double word of each line is accessed. This creates simultaneous contention for multiple cache lines rather than a single line.
cons4	Similar to cons2, except that each CPU performs one store byte (`storeb`) and one load byte (`loadb`) operation between the detection of ID and the write of the next CPU ID. The target of the `storeb` and `loadb` is a unique byte in the line the CPU just read. This target is recognized as a different double word in the shared line *cachesize* bytes.
cons5	Similar to cons3 except that each CPU performs one `storeb` and one `loadb` operation between the detection of ID and the write of the next CPU ID. The target of the `storeb` is one unique byte of the next double word in the line that the CPU just read from the CPU ID. The `storeb` data is unique to each CPU and changes each time the address of the target line changes.
cons6	Similar to cons1 except that only one double word of each line is accessed. This creates simultaneous contention for multiple cache lines rather than a single line.
cons7	Similar to cons3 except that each CPU performs two `storeb` and one `loadh` operations between the detection of the CPU ID and the write of the next CPU ID. The targets of the `storebs` and `loadh` are two consecutive bytes of a double word in a shared line which is not a part of the shared memory buffer containing the IDs. The address of the target `storeb` and `loadh` instructions is held constant. The first `storeb` instruction gains ownership of the cache line, and the second `stroreb` is performed as a write hit. This occurs at the same time other CPUs are reading and writing the shared line containing the IDs.

TABLE 37-1 `mpconstest` subtests

Tests	Description
cons8	Similar to cons3 except that each CPU performs one `storeb` and one `loadb` operation between the detection of the CPU ID and the write of the next CPU ID. The target of the `storeb` and `loadb` is one unique byte of a double word of a private (unshared) line whose line number is identical to the line number containing the IDs. The `storeb` data is unique to each CPU and changes each time the address of the line containing the IDs changes.
cons9	Similar to cons8 except that the target of the `storeb` and `loadb` is one unique byte of a double word of a private line whose address does not change through the entire test.
cons10	Similar to cons9 except that two `storeb` and two `loadb` operations are performed to private (unshared) lines.The target of the second `storeb` is *cachesize* bytes away from the target of the first `storeb`. In a direct map cache, this results in a writeback of the unshared data written with the first `storeb`. The `loadb` operations are performed after the `storeb` in order to ensure that the writeback occurs correctly.
cons11	Similar to cons10 except that the target of the `storeb` and `loadb` operations is to a shared line rather than a private line.
cons12	Similar to cons7 except that two store double (`stored`) and load double (`loadd`) operations are used in place of the `storeb` and `loadb` operations. The target of the `stored` and `loadd` operations are two consecutive double words of a shared line. This test is designed to verify that the double word operations are performed correctly while the shared and owned state of the line containing the ID is changing.
cons13 through cons17	These tests are similar variations of intermediate operations, stride size etc, and do not involve any new interfaces.

`mpconstest` Options

To reach the 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 may not include the device appropriate to this test. Refer to the *SunVTS User's Guide* for more details.

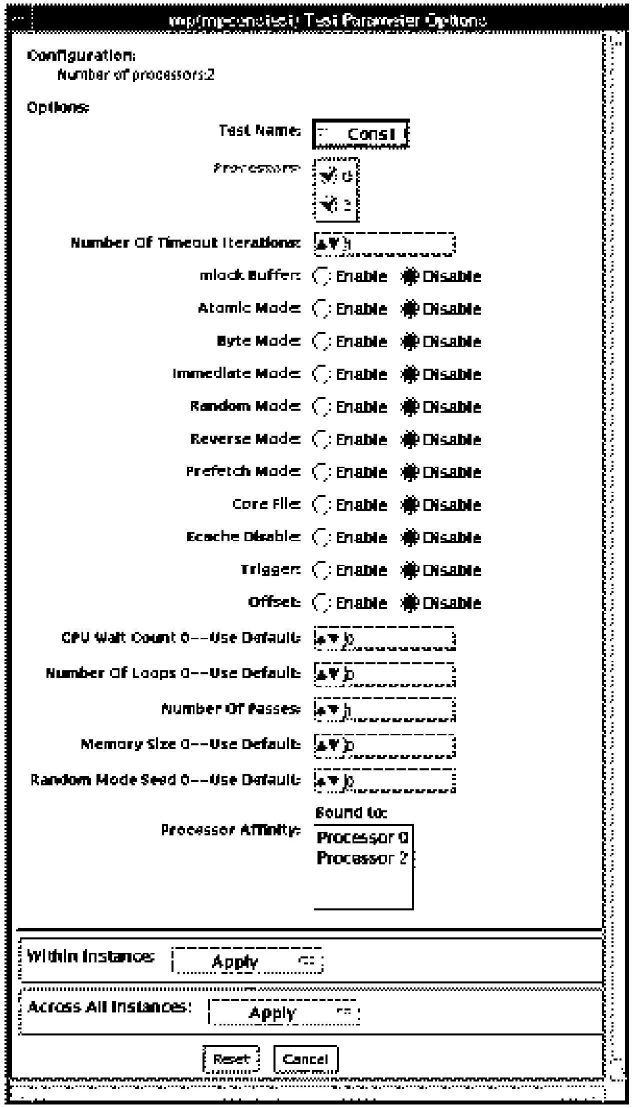


FIGURE 37-1 `mpconstest` Test Parameter Options Dialog Box

TABLE 37-2 `mpconstest` Options

Option	Description
Test Name	Selects the subtest to be run.
Number of Timeout Iterations	Sets the number of times the test is allowed to time out. Default is 1. Note that each timeout occurs after a greater amount of elapsed time than the previous one. That is, if the first timeout occurs after T units of time, the second occurs 2T after T, and the third occurs 3T after 2T.
Lock Buffer	Locks Buffer in Memory. Default is not locked. Locking the buffer in memory will disable COMA (Cache Only Memory Architecture).
Atomic Mode	Uses the atomic instruction swap. Default is disabled.
Byte Mode	Uses byte instructions to load and store. Default is disabled.
Immediate Mode	Supports all subtests except cons1, cons2, cons3, cons15, cons16, and cons17.
Random Mode	Enables Random Mode.
Reverse Mode	Traverses the shared memory buffer in reverse. Default is disabled.
Prefetch Mode	Sets prefetch for read and write. Default is disabled.
CoreFile	Generates a core file. Exits in case of unexpected signals. Default is disabled.
Ecache Disable	Disables the external cache. Default is enabled.
Trigger	Sends an interrupt signal to all processors when one processor detects a failure. Default is disabled.
Offset	Specifies an offset of line size between successive writes. Default is disabled.
CPU Wait Count	Forces CPU 1 to write first if the number of CPUs is less than *cpucount*. Default is disabled. This option is not supported with subtests cons15, cons16, and cons17.
Number of Loops	Selects the number of test loops. Default is 5.
Number of Passes	Selects the number of passes. Increasing the number of passes increases system stress. Setting the number of passes to 0 will cause the test to run in an endless loop. Passes can only be set to 0 in command line mode, not from the GUI. Default is 1.
Memory Size	Selects the memory size, in Megabytes, for the shared buffer. Default is 128.
Random Mode Seed	Sets random number seed to a user specified value. Selects a random number seed by default.

`mpconstest` Test Modes

TABLE 37-3 `mpconstest` Test Modes

Test Mode	Supported?	Description
Connection	No	Not supported.
Functional (Offline)	Yes	Runs the full test.

`mpconstest` Command-Line Syntax

`/opt/SUNWvts/bin/mpconstest` *standard_arguments*
`-o`
`tst=`*Cons1* | *Cons2***`,itm=`***number***`,lck,a,b,c,e,h,loops=`***number***`,memsize=`***memsize***`,`**
`wait=`*cpucount***`,passes=`***passes***`,r,t,x,y,i,q,seed`**=*number*

TABLE 37-4 `mpconstest` Command-Line Syntax

Argument	Description
`tst=`*Cons1* | *Cons2* | *Cons3* | *Cons4* | *Cons5* | *Cons6* | *Cons7* | *Cons8* | *Cons9* | *Cons10* | *Cons11* | *Cons12* | *Cons12* | *Cons13* | *Cons14* | *Cons15* | *Cons16* | *Cons17*	Range of choices available between cons1 through cons17 subtests.
`itm=`*number*	Sets the number of times the test is allowed to time out. Default is 1. Note that each timeout occurs after a greater amount of elapsed time than the previous one. That is, if the first timeout occurs after `T` units of time, the second occurs `2T` after `T`, and the third occurs `3T` after `2T`.
`lck`	Locks Buffer in Memory. Default is not locked. Locking the buffer in memory will disable COMA (Cache Only Memory Architecture).
`a`	Enables atomic mode. Uses the atomic instruction swap
`b`	Enables byte mode. Uses byte instructions to load and store.

TABLE 37-4 `mpconstest` Command-Line Syntax

Argument	Description
`c`	Generates a core file. Exits in case of unexpected signals.
`e`	Disables the external cache.
`h`	Prints usage message.
`loops=`*number*	Sets the number of loops for the iterations. Default is 5.
`memsize=`*memsize*	Selects the memory size, in Megabytes, for the shared buffer. Default is 128.
`wait=`*cpucount*	Forces CPU 1 to write first if the number of CPUs is less than *cpucount*.
`passes=`*passes*	Selects the number of passes. Increasing the number of passes increases system stress. Setting the number of passes to 0 will cause the test to run in an endless loop. Passes can only be set to 0 in command line mode, not from the GUI. Default is 1.
`r`	Enables Reverse mode. Traverses the shared memory buffer in reverse.
`t`	Enables Trigger. Sends an interrupt signal to all processors when one processor detects a failure.
`x`	Enables Prefetch. Sets prefetch for read and write.
`y`	Enables Offset. Specifies an offset of line size between successive writes.
`i`	Enables Immidiate Mode. Not suppored for subtests cons1, cons2, cons3, cons15, and cons 17.
`q`	Enables Random Mode.
`seed`	Sets a random number seed to the user specified value.

CHAPTER 38

Multiprocessor Test (mptest)

mptest verifies the functionality of multiprocessing hardware. This test allocates a page of virtual memory for the test—declaring the page shared—locks the page against swapping, and creates threads to each of the processors being tested. Up to 1024 processors can be tested by mptest in a CPU.

The *processor mask* argument can used during test probing. The mptest verifies that the current processor mask matches the argument you entered in the command line or from the GUI/TTYUI.

mptest Options

To reach the 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 may not include the device appropriate to this test. Refer to the *SunVTS User's Guide* for more details.

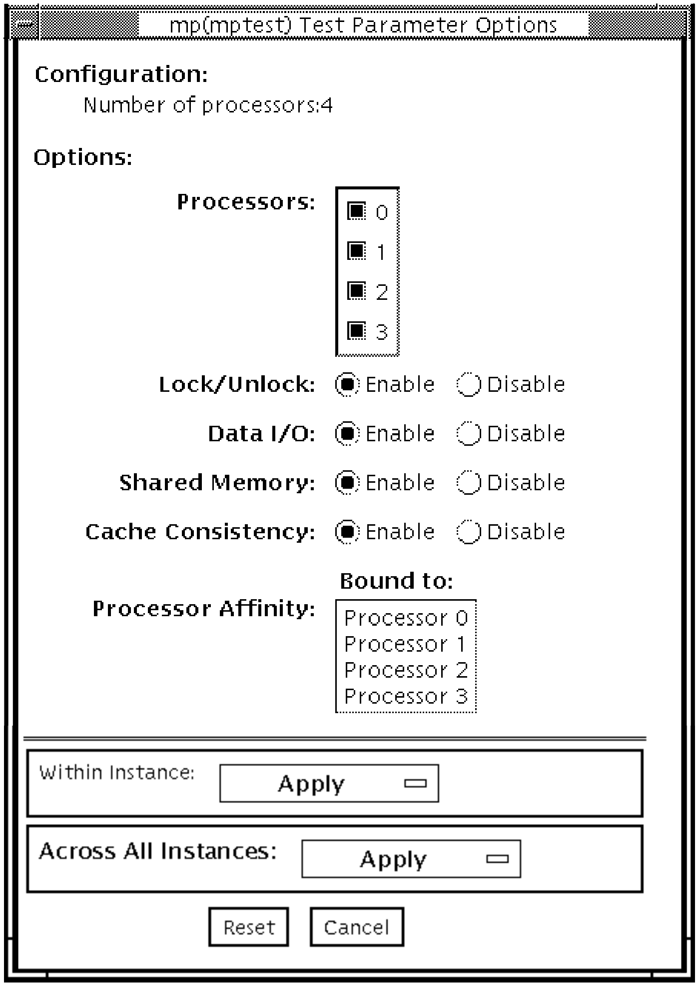


FIGURE 38-1 `mptest` Test Parameter Options Dialog Box

The processors that can be tested are listed in the Configuration area of the menu. You can enable or disable the multiprocessing test for individual processors on this menu.

The options listed in TABLE 38-1 can be run alone or concurrently with other options.

TABLE 38-1 `mptest` Options

`mptest` **Options**	**Description**
Processors	You can test specific processors by clicking Select on the check boxes to enable or disable each processor. A check mark means the processor is enabled for testing. The default setting is all processors enabled. Note: `mptest` requires at least two enabled processors to test multiprocessing systems.
Lock / Unlock	Tests the lock / unlock mechanism that guarantees exclusive access to a physical page to one processor. A thread is created at each of the processors. Each processor uses the SPARC atomic instruction `ldstub` to write to the same shared physical memory page. While one processor is attempting the write, the other processors should be free spinning for their turn. As each processor acquires the lock, it writes an ordinal number to a shared trace buffer using a shared write pointer. After the test cycle is complete, the trace buffer is dumped for analysis. This test fails and returns an error message if the trace buffer does not contain an equal number of ordinal numbers for each processor. For example, if the specified loop count is 5, the trace buffer should contain five 0s, five negative 1s, five 2s, and so on.

TABLE 38-1 `mptest` Options *(Continued)*

`mptest` **Options**	**Description**
Data I/O	Requires two or more threads, each of which locks onto one of the processors. Each processor, in turn, writes data to a temporary file that has been mapped to the physical address. The modified data is immediately read by other processors being tested. This test hangs and fails if the processors do not recognize the expected data.
Shared Memory	A shared memory buffer is divided into a number of contiguous chunks, one for each of the CPUs participating in the test. Each CPU is assigned a unique chunk based upon its ID (1-N). This subtest has two parts. First, each CPU locks and writes data to its data chunk. Identical data is written for each CPU. Then each CPU reads and compares the information on its data chunk with that of another CPU. If two CPUs do not confirm consistent data, the test fails and returns an error message. If that happens, testing stops and this test is run again in verbose mode to return more detailed information.
Cache Consistency	Requires two or more processors to access and write to the same physical address. This test verifies that a change in physical address by one processor is confirmed by another. If two processors do not confirm consistent data, the test continues to run, but the Pass Count in the SunVTS status window stops incrementing. If this happens, stop testing and run the test again in verbose mode for a more detailed picture of the problem.

mptest Test Modes

TABLE 38-2 mptest Test Modes

Test Mode	Supported?	Description
Connection	Yes	Checks the current processors on the system with the original processor mask. An error is reported if the two values do not match. The original processor mask is set during probing, which shows the processors on system during the probe. The status of each selected processor is checked by `procsesor_bind`.
Functional (Offline)	Yes	This test mode verifies that the current processor mask is the same as that from the command line, or the same as that from the GUI/TTYUI.

mptest Command-Line Syntax

/opt/SUNWvts/bin/mptest *standard_arguments*
-o M=0+1+2+3...,NL,ND,NS,NC,omask=*hexidecimal_number*

TABLE 38-3 mptest Command-Line Syntax

Arguments	Description
M=*0+1+2+3...*	Use *0*, *1*, *2* to specify the processors to test.
NL	Disables the Lock/Unlock subtest.
ND	Disables the Data I/O subtest.
NS	Disables the Shared Memory subtest.
NC	Disables the Cache Consistency subtest.
omask=*hexidecimal_number*	Original mask of processors. Bit 0 represents processor 0 and bit 1 represents processor 1. For example, `03333320`.

Note – 64-bit tests are located in the `sparcv9` subdirectory: `/opt/SUNWvts/bin/sparcv9/`*testname*. 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 “32-Bit and 64-Bit Tests” on page 3.

CHAPTER **39**

Network Hardware Test (nettest)

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

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 it fails to find all necessary 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 200. So, 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 200 instances per each networked device.

`nettest` Options

To reach the 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 may not include the device appropriate to this test. Refer to the *SunVTS User's Guide* for more details.

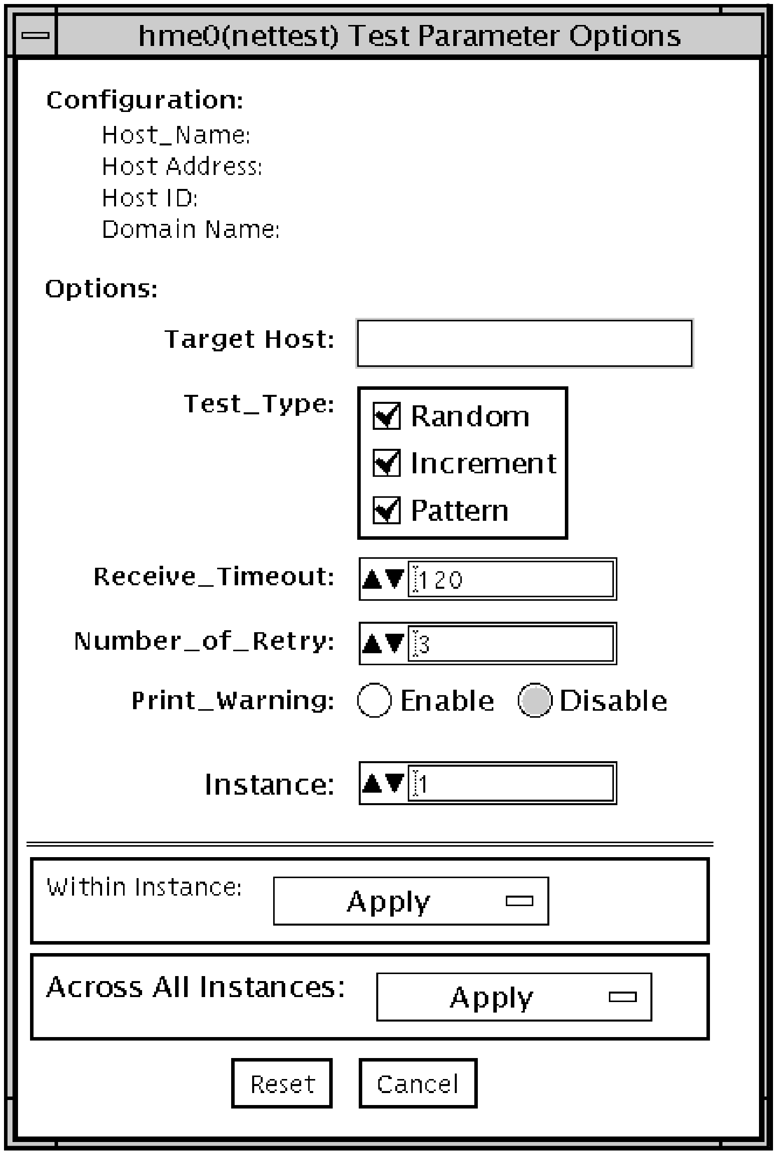


FIGURE 39-1 `nettest` Test Parameter Options Dialog Box

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

TABLE 39-1 `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

All three modes are supported by `nettest`. Different test schemes are performed on the network device based on the mode selected.

TABLE 39-2 `nettest` Test Modes

Test Mode	**Supported?**	**Description**
Connection	Yes	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 (Offline)	Yes	Performs all three tests (Random test, Incremental test, and Pattern test) sequentially. It allows you to specify options that will perform heavy stress testing.

`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 39-3 `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. Type `Random`, `Increment`, or `Pattern` for the desired test. The default value is Random+Increment+Pattern where all tests run.
`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*. 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 “32-Bit and 64-Bit Tests” on page 3.

CHAPTER **40**

Ethernet Loopback Test (`netlbtest`)

The `netlbtest` replaces the `gemtest` previously included in SunVTS. It provides functional test coverage of the devices whose device drivers support the Ethernet loopback test. These include `eri` (the Ethernet device in the RIO chip) and `ge` (Gigabit Ethernet), and `ce` (GigaSwift Ethernet) . It runs in loopback (external/ internal) mode.

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 the packet sent is selected from a different starting point of the data buffer, so that any two consecutively transmitted packets will not be the same.

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

`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`. Even though the `netlbtest` does not depend on the network interface configuration status of the Ethernet interface, the interface should be brought down with the `ifconfig` command (refer to the `ifconfig(1M)` man page) to avoid erroneous messages.

To run `netlbtest`, a loopback connector must be connected to the Ethernet interface. `netlbtest` cannot run if the network interface is connected to a live network, however the link must be up. 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` 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 RJ45 connector. See "Twisted-Pair Ethernet (TPE) Loopback Cable" on page 389 for the diagram.

`netlbtest` also requires that the `eri` device be configured offline before running the test. Type the following two commands at the command prompt:

```
# ifconfig ge0 down
# ifconfig ge0 unplumb
```

`netlbtest` Options

To reach the 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 may not include the device appropriate to this test. Refer to the *SunVTS User's Guide* for more details.

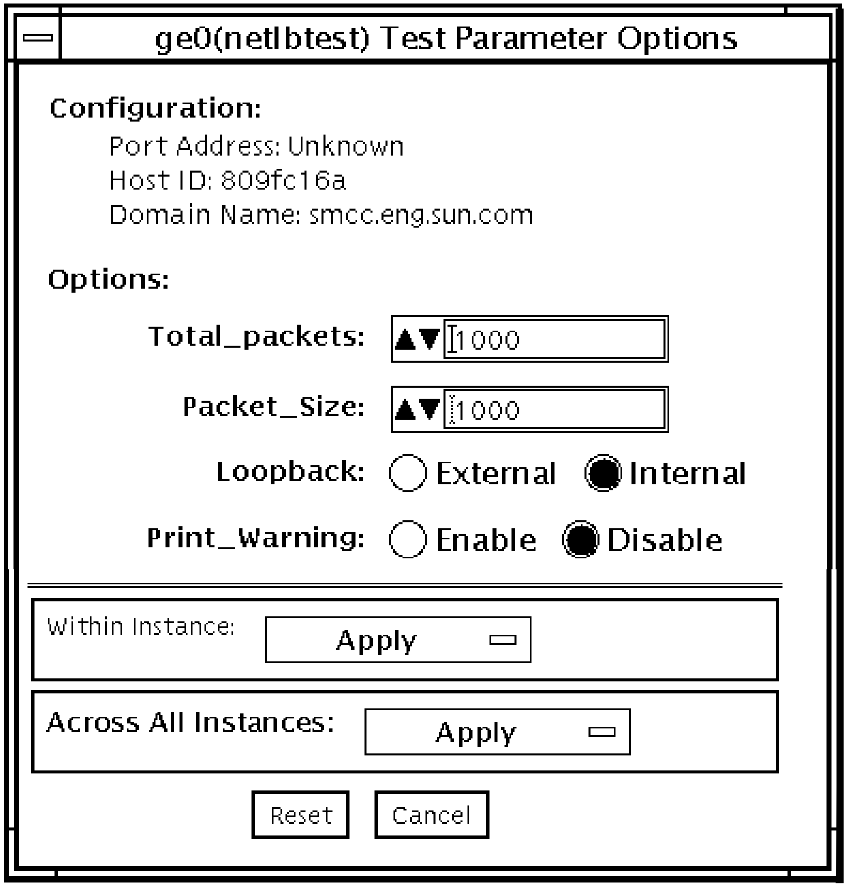


FIGURE 40-1 `netlbtest` Test Parameter Options Dialog Box

Refer to TABLE 40-1 for test parameter descriptions.

TABLE 40-1 `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 packet is 1000.
Packet size	Determines the size (in bytes) of the packets to be transmitted. 60 <= packet size <= 1514. The default packet size is 1000 bytes.

TABLE 40-1 `netlbtest` Options

`netlbtest` **Options**	**Description**
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.

`netlbtest` Test Modes

TABLE 40-2 `netlbtest` Test Modes

Test Mode	**Supported?**	**Description**
Connection	No	Not supported.
Functional (Offline)	Yes	Runs the full set of subtests. It is assumed that the host is not connected to the network through the intended test device(s).

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*

TABLE 40-3 `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.

TABLE 40-3 `netlbtest` Command-Line Syntax

Argument	Description
pksz=*pkt_size*	[60... 1514], packet size in bytes.
lb=*Internal*	Selects internal (or external) loopback mode.
warn=*Disable*	Enables or disables printing of warning messages.

Note – 64-bit tests are located in the `sparcv9` subdirectory: `/opt/SUNWvts/bin/sparcv9/`*testname*. 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 “32-Bit and 64-Bit Tests” on page 3.

CHAPTER **41**

PCMCIA Modem Card Test (pcsertest)

PCMCIA is a technology that provides small, easy to use peripheral devices. PCMCIA stands for Personal Computer Memory Card International Association. It is a PC Card standard for mobile computing I/O cards. These cards range from memory, FAX/modems, serial I/O, SCSI, video, sound, and so on.

The `pcsertest` verifies the functionality of PCMCIA modem card and PCMCIA serial I/O socket card. It does not test any other PCMCIA devices.

The `pcsertest` issues a series of commands to the modem to initiate a local analog loopback test and verifies this functionality.

As an option, the `pcsertest` tests serial I/O socket cards. This test writes a pattern of incrementing data to the serial I/O socket card, which is then looped back, read and verified.

Note – When testing serial I/O socket cards, a 9-pin loopback connector is required. However, no loopback connector is required when testing the default modem card. See Appendix A for loopback connector wiring instructions.

---

pcsertest Options

To reach the 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 may not include the device appropriate to this test. Refer to the *SunVTS User's Guide* for more details.

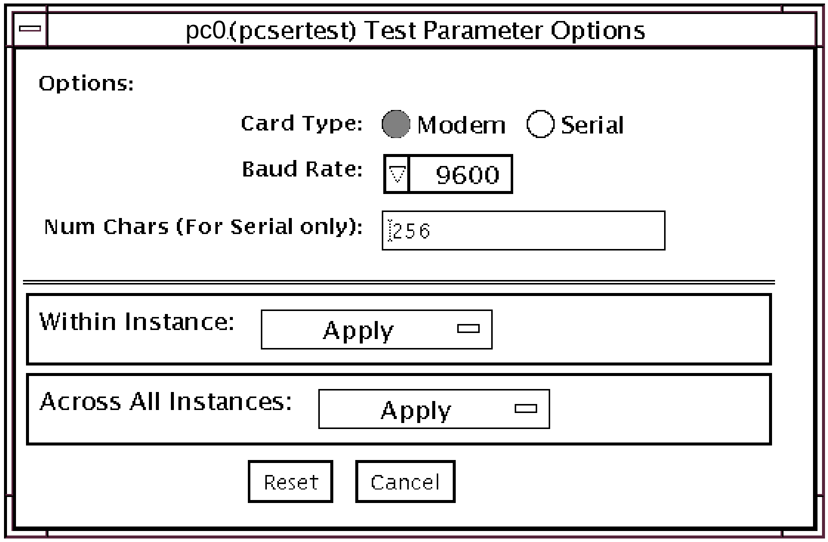


FIGURE 41-1 `pcsertest` Test Parameter Options Dialog Box

TABLE 41-1 `pcsertest` Options

`pcsertest` **Option**	**Description**
Card Type	Specifies the type of device you are testing; either a PCMCIA modem, or a PCMCIA serial I/O socket card. Note–If you choose the Serial card type, and there is no serial I/O socket card in the slot, the test fails.
Baud Rate	Specifies the baud rate for testing.
Num Chars	Specifies the number of characters being used for external loopback testing of the serial socket card. By default, this is set to 256 characters. This option is applicable only to serial socket cards and ignored for the modem card.

Note – Any combination of modem and socket I/O cards can be placed in the PCMCIA slots. However, you must select the correct type of card in the Options dialog box. If you select an incorrect card type, the test fails. The default card type for each PCMCIA slot is a modem card. If only one modem card is plugged in, the empty slot is ignored.

`pcsertest` Test Mode

TABLE 41-2 `pcsertest` Test Modes

Test Mode	Supported?	Description
Connection	No	Not supported.
Functional (Offline)	Yes	Runs the full set of tests.

`pcsertest` Command-Line Syntax

`/opt/SUNWvts/bin/pcsertest` *standard_arguments* **`-o dev=`***device_name***`,`**
`type=`*card_type***`,baudrate=`***speed***`,numchars=`***n*

TABLE 41-3 `pcsertest` Command-Line Syntax

Argument	Description
`dev=`*device_name*	Specifies the device name (for example, **`dev=pc0`** and **`dev=pc1`**)
`type=`*card_type*	Specify one of the two card types for the device (**`type=serial`** or **`type=modem`**). You do not need to specify the type if the device is a modem, since modem is the default card type.

`baudrate=`*speed*	Specifies the communication speed. Specify one of the following:
	`9600`
	`19200`
	`38400`
	`57600`
	The default is `9600`.

`numchars=`*n*	Specifies the number of characters to use for external loopback testing of the serial socket card. By default, this is set to `256` characters. This option is applicable only to serial socket cards and ignored for the modem card.

Note – 64-bit tests are located in the `sparcv9` subdirectory: `/opt/SUNWvts/bin/sparcv9/`*testname*. 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 “32-Bit and 64-Bit Tests” on page 3.

CHAPTER **42**

SPARCstorage Array Controller Test (plntest)

`plntest` checks the functionality of the controller board on the SPARCstorage™ Array. The SPARCstorage Array (SSA) is a large disk storage I/O subsystem capable of housing up to 30 SCSI hard drives. The SSA communicates with a host system over a fiber-optic link provided by an SBus-based host adapter card in the host system and the corresponding SSA controller board hardware.

The SSA controller card is an intelligent, CPU-based board with its own memory and ROM-resident software. In addition to providing a communications link to the disk drives, it also buffers data between the host system and disk drives in its nonvolatile RAM (NVRAM). For data to go from the host to a particular disk, it must first be successfully transferred to this NVRAM space.

The host machine, SBus host adapter card, Fibre Channel connection, and the SSA controller board must be working properly to perform this data transfer operation. By verifying and stressing this operation, `plntest` can isolate failures on the SSA disk drives from failures on the SSA controller board.

Note – `disktest` transfers data on the SSA disk drives over the same path mentioned above. However, `disktest` does not transfer data as quickly as `plntest`.

---

plntest Controller Test

The `plntest` exercises the hardware and software by invoking SCSI read buffer commands of various sizes to the NVRAM. These operations exercise the host Fibre Channel hardware, the SSA Fibre Channel hardware, the SSA resident management software, and the hardware component interaction on the SSA controller card (all components except the SCSI devices). In addition, the `plntest` reports failure of the fan module and the NVRAM battery module of the SPARCstorage Array.

Probing for SSA Controller Devices

Unlike most other hardware devices, the SSA controller card does not have a logical device name (one you would find in the `/dev` directory). Therefore, the SSA controller card is identified by its longer, physical device name.

Note – The physical device name of the SSA controller card cannot be used to run `plntest`.

When running `plntest` from the command line, the physical device name of the SSA controller card cannot be used, so a logical name must be specified. ANSI standards require commas as delimiters between items. Since the physical name of the SSA controller contains embedded commas, if you use a physical name that contains commas as a command-line option, `plntest` misinterprets the option.

There are two ways that you can create a logical name:

- Run the SunVTS kernel (`vtsk`), which automatically creates a logical name entry for the SSA controller under the `/dev` directory, such as:

```
# /dev/ssaXX , where XX represents the decimal number of the
controller
```

Use this name as the parameter for the **dev=** option of `plntest`.

- Manually make a soft link from the actual physical device name to a logical name of your choice (under `/dev`). Use this name as the parameter for the **dev=** option of `plntest`, as shown in the following example:

```
machine# ./plntest -o “?”

1: /devices/io-unit@f,e3200000/sbi@0,0/SUNW,soc@1,0/SUNW,
pln@0c0d,0e0f0102:ctlr
2: /devices/io-unit@f,e0200000/sbi@0,0/SUNW,soc@3,0/SUNW,
pln@0c0d,0e0f0102:ctlr

machine# ln -s \ /devices/io-unit@f,e3200000/sbi@0,0/SUNW,soc@1,0/SUNW,
pln@0c0d,0e0f0102:ctlr \ /dev/ssa1
machine# /opt/SUNWvts/bin/plntest dev=/dev/ssa1
```

`plntest` Options

To reach the 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 may not include the device appropriate to this test. Refer to the *SunVTS User’s Guide* for more details.

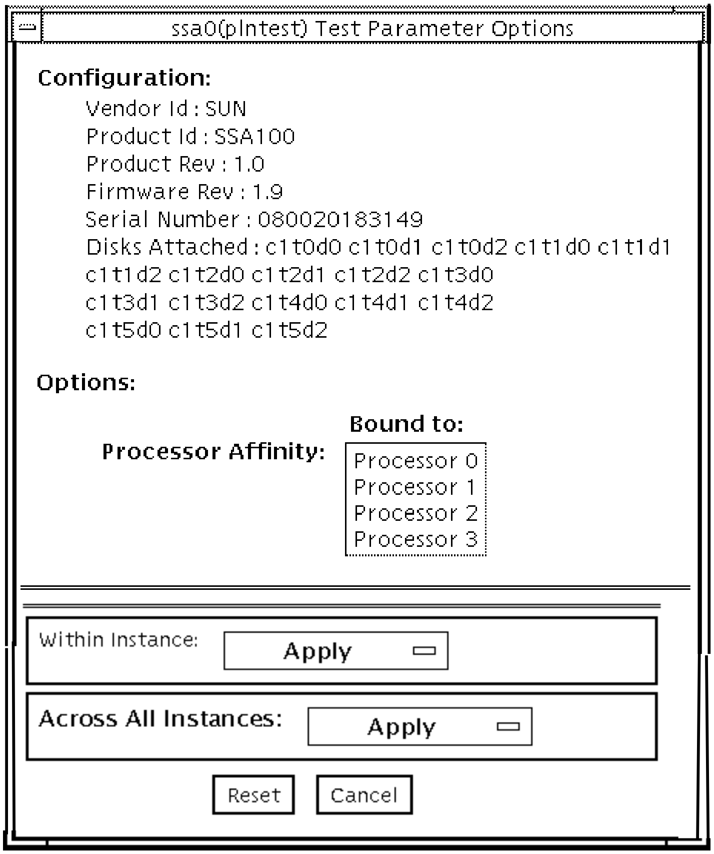


FIGURE 42-1 `plntest` Test Parameter Options Dialog Box

Configuration lists the names of all the logical disk drives (both single and grouped) that are attached to the SPARCstorage Array controller board. `plntest` is not a scalable test.

Note – If no disks are present, none is displayed under the Attached Disks heading.

---

plntest Test Modes

TABLE 42-1 plntest Test Modes

Test Mode	Supported?	Description
Connection	Yes	In this mode, plntest checks the state of the battery module and the fan module of the SPARCstorage Array.
Functional (Offline)	Yes	plntest checks the state of the battery module and the fan module of the SPARCstorage Array. The plntest issues a SCSI command, READBUFFER, causing the entire NVRAM to be read. This test uses different transfer buffer sizes.

plntest Command-Line Syntax

/opt/SUNWvts/bin/plntest *standard_arguments* **-o ?,dev=***device_name***,x**

TABLE 42-2 plntest Command-Line Syntax

Argument	Description
?	Probes the system for valid SSA controller devices and prints them to screen (see “Probing for SSA Controller Devices” on page 250).
dev=*device_name*	Specifies the physical path name of the SSA controller card to be tested. This argument *must* be included when running plntest from the command line, unless the **?** argument is used.
x	Probes the specified SSA controller card for the single and grouped disks attached to the controller card, and prints their logical names to the screen. Note: The **dev=***device_name* option must be specified for this option to work.

Note – 64-bit tests are located in the sparcv9 subdirectory: /opt/SUNWvts/bin/sparcv9/*testname*. 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 “32-Bit and 64-Bit Tests” on page 3.

CHAPTER 43

Physical Memory Test (pmemtest)

The pmemtest checks the physical memory of the system. The pmemtest locates parity errors, hard and soft error correction code (ECC) errors, memory read errors, and addressing problems. The pseudo driver mem(7) is used to read the physical memory.

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

pmemtest Options

To reach the 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 may not include the device appropriate to this test. Refer to the *SunVTS User's Guide* for more details.

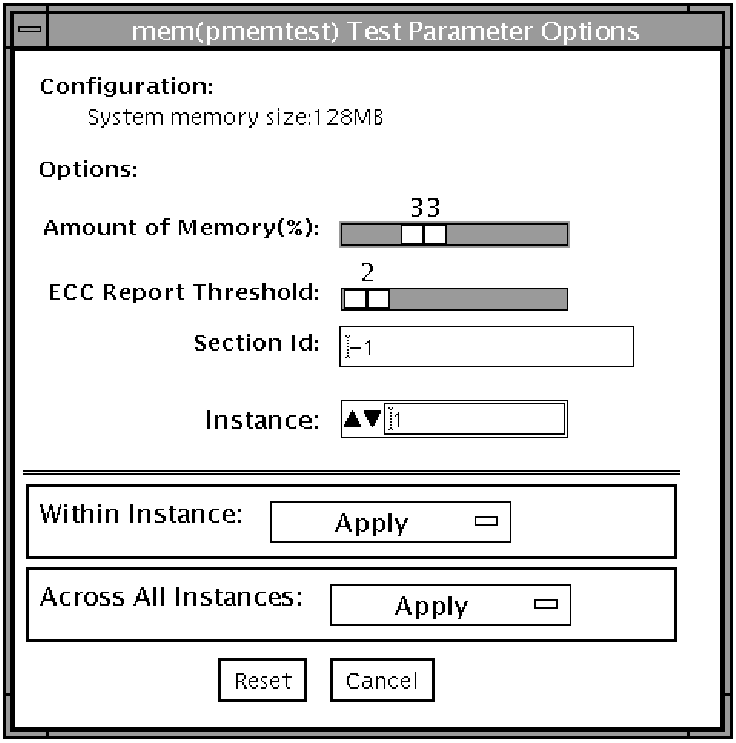


FIGURE 43-1 `pmemtest` Test Parameter Options Dialog Box

TABLE 43-1 `pmemtest` Options

`pmemtest` **Options**	**Description**
Configuration	Shows the total amount of physical memory, rounded up to the nearest megabyte, probed by the SunVTS kernel.
Amount of Memory	Specifies the percent of the physical memory to be tested. The default is 33%.

TABLE 43-1 `pmemtest` Options

`pmemtest` **Options**	**Description**
ECC Report Threshold	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. This option is only available on UltraSPARC™ systems.
Section ID	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 will be tested. A section is defined by the pass and instance number settings. This option is only available on UltraSPARC systems.
Instance	The number of copies of `pmemtest` to run simultaneously on the memory you are testing.

pmemtest Test Modes

TABLE 43-2 pmemtest Test Modes

Test Mode	Supported?	Description
Connection Test	Yes	In this mode, one percent of the memory is read. `pmemtest` also informs the user 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 (Offline)	Yes	In Functional test mode, the amount of memory to be read can vary. For UltraSPARC servers, this test mode reports all ECC errors that have occurred since the system was brought online

pmemtest Command-Line Syntax

/opt/SUNWvts/bin/pmemtest *standard_arguments*
-o size=*[1-100]***,dev=***device_name***,threshold=***report_threshold*
,bdinfo=*number***,section=***section_id*

TABLE 43-3 pmemtest Command-Line Syntax

Argument	Description
size=*[1-100]*	Specifies the percentage of memory to be tested. The default is 33%.
dev=*device_name*	Specifies the device to test, for example, `mem`.

TABLE 43-3 `pmemtest` Command-Line Syntax

Argument	Description
`threshold=`*report_threshold*	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. This option is only available on UltraSPARC systems.
`bdinfo=`*number*	For UltraSPARC servers, 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 bdinfo=33 (2**5+2**0).
`section=`*section_id*	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 will be tested.A section is defined by the pass and instance number settings. This option is only available on UltraSPARC systems.

Note – 64-bit tests are located in the `sparcv9` subdirectory: `/opt/SUNWvts/bin/sparcv9/`*testname*. 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 “32-Bit and 64-Bit Tests” on page 3.

CHAPTER **44**

Qlogic 2202 Board Test (`qlctest`)

`qlctest` is made up of several subtests that test the functions of the Qlogic 2202 FC/AL Crystal. Unlike the earlier single-port Q2100 board, the Q2202 is a two-port board which has greater diagnostic support.

This test is not scalable.

Note – Do not run customer data while running `qlctest`, as the test will take priority over customer data requests. The customer will be unable to access data while `qlctest` is running.

Note – Do not run other tests while `qlctest` is running. `qlctest` may cause other tests to fail.

Note – `qlctest` is an intervention mode test. No subtests can be selected unless intervention is set.

`qlctest` Subtests

There are nine possible subtests to run in intervention and functional modes:

- Fcode revision check
- Firmware revision check
- Board revision check
- Checksum Firmware subtest
- Selftest

- Mailbox Loopback subtest
- Internal 10-bit Loopback subtest
- Internal 1-bit Loopback subtest
- External Loopback subtest

The external loopback test is an intervention test. To test the fibre loop, leave the QLC port attached to the storage. In the Test Parameters Options dialog box, set the "Test if Connected to Storage" option to "Yes". To test the Qlogic 2202 board alone, connect a loopback cable to the QLC port. This cable can be made by taking a regular cable and splitting it apart. Then loop the transmitter side of the port to the receiver side of the port.

For subtest descriptions, see TABLE 44-1 on page 263.

`qlctest` Options

To reach the 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 may not include the device appropriate to this test. Refer to the *SunVTS User's Guide* for more details.

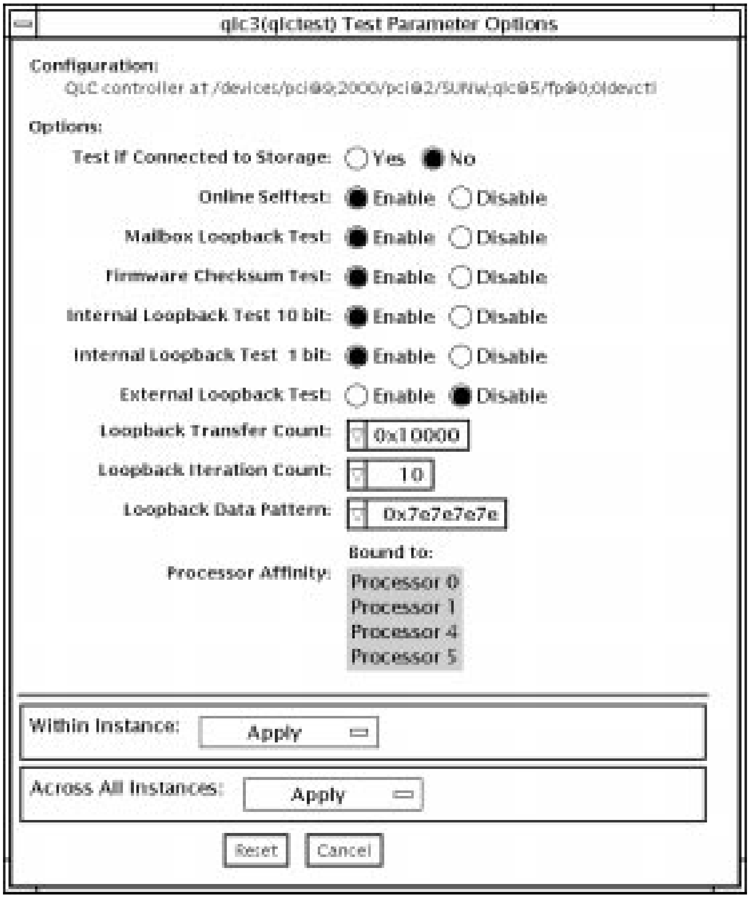


FIGURE 44-1 `qlctest` Test Parameter Options Dialog Box

TABLE 44-1 `qlctest` Options

`qlctest` **Options**	**Description**
Fcode revision check	Retrieves the fcode revision string. A core subtest that is always run but not shown in the Options dialog box.
Firmware revision check	Retrieves the firmware revision string. A core subtest that is always run but not shown in the Options dialog box.
Board revision check	Retrieves the board revision levels. A core subtest that is always run but not shown in the Options dialog box.
Test if Connected to Storage	Runs `qlctest` while connected to storage. Default value is No.

TABLE 44-1 `qlctest` Options

`qlctest` **Options**	**Description**
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.
Mailbox Loopback subtest	Loads a series of registers into the input mailboxes on the card and then reads the output mailboxes and compares results. This verifies that the system side of the card is operating correctly, and that the internal data paths are correct. Run by default, but can be deselected.
Firmware Checksum subtest	Runs an internal checksum test on the installed firmware. This verifies that the RISC RAM on the card is fully functional and that the installed firmware is still intact. This test also serves as a quick RAM check of the RISC RAM. Run by default, but can be deselected.
Internal 10-bit Loopback subtest	Performs internal loopback test within the host adapter ISP hardware at the 10-bit 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 via the test parameters menu. Run by default, but can be deselected.
Internal 1-bit Loopback subtest	Performs internal loopback test within the host adapter ISP hardware at the 1-bit 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 via the test parameters menu. Run by default, but can be deselected.
External Loopback subtest	Performs an external loopback test. 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 via the test parameters menu. This is an intervention test, because a loopback cable is needed from the transceiver to the receiver of the QLC 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.

TABLE 44-1 `qlctest` Options

`qlctest` **Options**	**Description**
Loopback Transfer Count	Controls the packet size used in the internal 10-bit, internal 1-bit, and external loopback tests. Default value is `0x10000`.
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`.

qlctest Test Modes

TABLE 44-2 qlctest Test Modes

Test Mode	Supported?	Description
Connection	Yes	Opens and closes the QLC port.
Functional (Offline)	Yes	Runs the full set of tests.

qlctest Command-Line Syntax

/opt/SUNWvts/bin/qlctest *standard_arguments*
-v -o dev=*device name***,run_connect=***Yes | No***,selftest=***Enable | Disable***,**
mbox=*Enable | Disable***,checksum=***Enable | Disable***,ilb_10=***Enable | Disable***,**
ilb=*Enable | Disable***,elb=***Enable | Disable***,xcnt=***0xtransfer_count***,**
icnt=*iteration_count***, lbfpattern=***0xpattern*

TABLE 44-3 qlctest Command-Line Syntax

Argument	Description
dev	The name of the device to test.
run_connect=*Yes | No*	If **run_connect** is set to Yes, qlctest will run when the tested port is connected to storage. If the port being tested is not connected to storage, this option has no effect. Default value is No.
selftest= *Enable | Disable*	Enables or disables the selftest command. Evaluates the functionality of the ISP hardware. Enabled by default.
mbox=*Enable | Disable*	Enables or disables the mailbox loopback command. This test writes data patterns into the mailboxes and then reads them back from the output mailboxes and verifies the data is correct. Enabled by default.
checksum= *Enable | Disable*	Enables or disables the checksum command. Runs an internal checksum test on the installed firmware. This verifies that the RISC RAM on the card is fully functional and that the installed firmware is still intact. This test also serves as a quick RAM check of the RISC RAM. Enabled by default.

TABLE 44-3 `qlctest` Command-Line Syntax

Argument	Description
`ilb_10=`*Enable | Disable*	Enables or disables the internal 10-bit test. Performs internal loopback test within the host adapter ISP hardware at the 10-bit interface. Enabled by default.
`ilb=`*Enable | Disable*	Enables or disables the internal 1-bit 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 via the test parameters menu. Requires a cable for this intervention test. Disabled by default.
`xcnt=`*0xtransfer_count*	Controls the packet size to be transferred, for example, 0x1000. Default value is 0x10000.
`icnt=`*iteration_count*	Controls the number of times the loopback test will run, for example, 100. Default value is 10.
`lbfpattern=`*0xpattern*	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*. 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 “32-Bit and 64-Bit Tests” on page 3.

CHAPTER 45

Remote System Control (`rsctest`)

The `rsctest` exercises the Remote System Control (RSC) feature, which is integrated on the Sun Enterprise 250 as well as the next-generation RSC 2.0 plug-in card introduced with the Sun Fire 280R line.

The RSC provides secure remote access for system monitoring, firmware updates, and failure recovery. The RSC communicates with the host through two internal serial lines, the I2C bus, and reset lines.

The RSC 1.0 hardware consists of the controller, flash, SEEPROM, 10MB Ethernet port, and an external console serial port.

The RSC 2.0 plug-in card hardware consists of the controller, flash, SEEPROM, 10MB Ethernet port, FRUSEEPROM, Time Of Day (TOD) device, internal PCMCIA modem card, and battery backup.

`rsctest` is not scalable.

`rsctest` Subtests

The `rsctest` will present different subtests and options based on which revision of the RSC hardware it is testing.

The subtests common to both RSC 1.0 and 2.0 include:

TABLE 45-1 Subtests for both RSC 1.0 and 2.0

Subtest	Description
Ethernet	Allows for internal loopback testing, on the Ethernet device with user specified data, size, and number of packets. Allows for external loopback testing with user-specified data, size, and number of packets. This requires a connection to a 10MB hub or switch for RSC 1.0, or a passive loopback connector for RSC 2.0. Allows for a ping to be sent to a specified host and checks the response.
Flash CRC	Performs a checksum test on the flash device.
SEEPROM CRC	Performs a checksum test on the SEEPROM device.
Serial	Allows internal loopback testing with user-specified data and size on the two internal serial ports. Allows for internal and/or external testing on the external `ttyu` port. The external test requires a passive loopback connector.

`rsctest` also presents the following subtests when running on the RSC 2.0 hardware:

TABLE 45-2 Subtests for RSC 2.0 Only

Subtest	Description
FRU SEEPROM CRC	Performs a checksum test on the SEEPROM device.
I2C	Tests the i2c bus connection between the host and the RSC.
TOD	Performs multiple reads to the TOD device and verifies that the time is incrementing.
Modem	Verifies that the modem is installed. Displays the manufacture information, in Verbose mode. Performs AT inquiry commands.

The subtests call test modlets that are written in the native Real Time Operating System (RTOS) that resides in the RSC firmware. The `rsctest` subtests execute the test modlets, passes parameters, and retrieves results from the RSC using a test protocol on the host to RSC internal serial lines.

`rsctest` Options

To reach the 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 may not include the device appropriate to this test. Refer to the *SunVTS User's Guide* for more details.

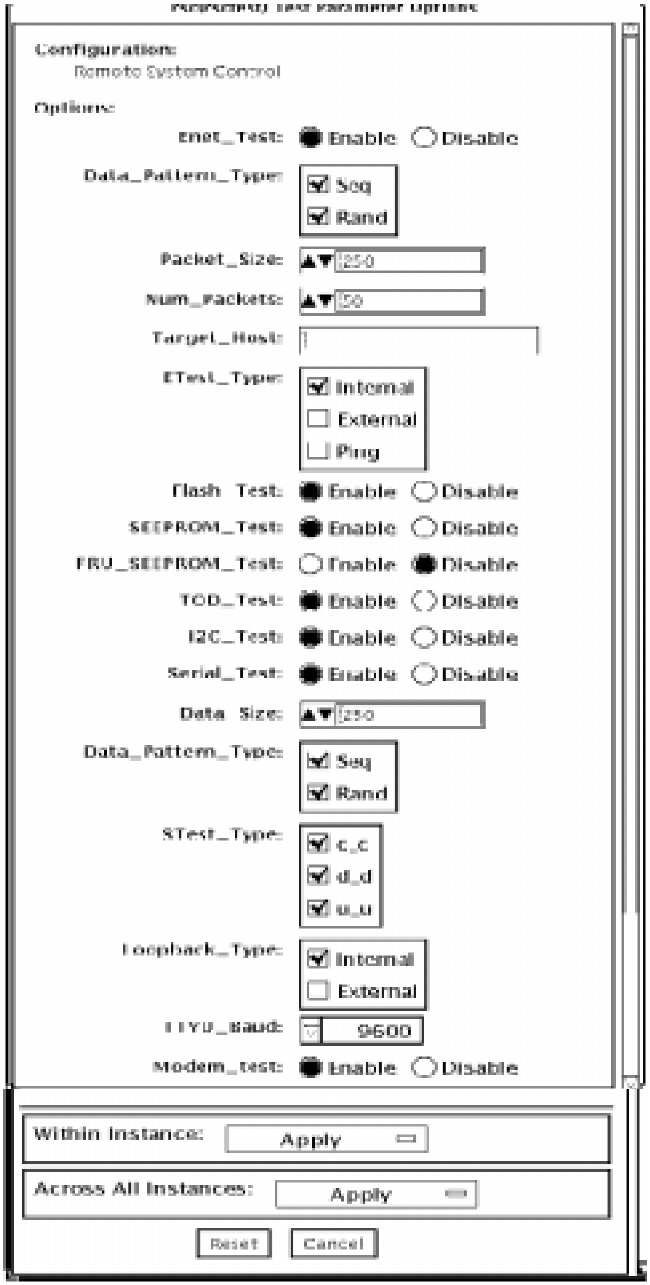


FIGURE 45-1 `rsctest` Test Parameter Options Dialog Box

TABLE 45-3 `rsctest` Options

`rsctest` **Options**	**Description**
Enet test	Enables or disables RSC Ethernet testing.
Data Pattern Type	Selects Sequential, Random, or both types of data patterns.
Packet Size	Defines the size of each data packet to be sent for all tests.
Num Packets	Specifies the number of data packets to send in one test loop.
Target Host	Specifies the IP address of a host to use for the ping test.
Enet Test Type	Selects any or all Internal, External, or ping tests.
Flash test	Enables or disables the flash checksum test.
SEEPROM test	Enables or disables the SEEPROM checksum test.
FRU SEEPROM test	Enables or disables the FRU SEEPROM checksum test (RSC 2.0 only).
TOD test	Enables or disables the Time Of Day test.
I2C test	Enables or disables the I2C test (RSC 2.0 only).
Serial test	Enables or disables the RSC serial test.
Data Size	Defines the data size to be sent.
Loopback Type	Selects Internal, External, or both. External requires a loopback plug.
Data Pattern Type	Selects Sequential, Random, or both types of data patterns.
Serial Test Type	Selects serial ports to be tested, u to u, c to c, or d to d.
TTYU_Baud	Selecta a fixed baud rate or all baud rates for testing the ttyu port.
Modem Test	Used to Enable or Disable the RSC PCMCIA modem test (RSC 2.0 only).

`rsctest` Test Modes

`rsctest` supports Connection and Functional tests as described in the table below.

TABLE 45-4 `rsctest` Test Modes

Test Mode	Supported ?	Description
Connection	Yes	Reports the status of the RSC.
Functional (Offline	Yes	Tests the RSC's Ethernet, flash, SEEPROM, and serial devices. All tests use internal modes as defaults. The `rsctest` will not run the serial test on ttyc if the console has been redirected to the RSC. The ttyu tests will not run if there is an open login on `ttyu`.

`rsctest` Command-Line Syntax

RSC 1.0: **`/opt/SUNWvts/bin/rsctest`** *standard_arguments* **`-o enet=`***E/D*, **`epattype=`***seq+rand*, **`esize=`***packet_size*, **`epkts=`***number_packets*, **`target=`***IP_address*, **`etest=`***I+E+P*, **`flash=`***E/D*, **`seeprom=`***E/D*, **`serial=`***E/D*, **`sdatsize=`***data_size*, **`slb=`***I+E*, **`spattype=`***seq+rand*, **`stest=`***u_u+c_c+d_d*, **`ttyubaud=`***baud_rate|all*

RSC 2.0: **`/opt/SUNWvts/bin/rsctest`** *standard_arguments* **`-o enet=`***E/D*, **`epattype=`***seq+rand*, **`esize=`***packet_size*, **`epkts=`***number_packets*, **`target=`***IP_address*, **`etest=`***I+E+P*, **`flash=`***E/D*, **`seeprom=`***E/D*, **`fruseeprom=`***E/D*, **`tod=`***E/D*, **`i2c=`***E/D*, **`serial=`***E|D*, **`sdatsize=`***data_size*, **`slb=`***I+E*, **`spattype=`***seq+rand*, **`stest=`***u_u+c_c+d_d*, **`ttyubaud=`***baud_rate|all*, **`rscmodem=`***E/D*

TABLE 45-5 `rsctest` Command-Line Syntax

Argument	Description
`enet=`*enable|disable*	Enables or disables RSC Ethernet test.
`epattype=`*seq+rand*	Predefined pattern options used for Enet test.
`esize=`*packet_size*	Data size for each packet in the Enet test.
`epkts=`*number_packets*	Number of packets to send for Enet test.
`target=`*IP_address*	IP address of target system for Enet ping test.
`etest=`*Internal+External+ Ping*	Selects any or all Internal, External, or ping tests.

TABLE 45-5 `rsctest` Command-Line Syntax

Argument	Description
`flash=`*enable* | *disable*	Enables or disables RSC Flash Checksum test.
`seeprom=`*enable* | *disable*	Enables or disables RSC SEEPROM checksum test.
`fruseeprom=`*E/D* (RSC 2.0 ONLY)	Enables or disables RSC FRU SEEPROM checksum test.
`tod=`*E/D (RSC 2.0 ONLY)*	Enables or disables RSC Time Of Day test.
`i2c=`*E/D* (RSC 2.0 ONLY)	Enables or disables RSC i2c test.
`serial=`*enable* | *disable*	Enables or disables RSC serial test.
`sdatsize=`*data_size*	Data size for the rsc serial tests.
`slb=`*Internal*+*External*	Loopback type. External N/A on ports C and D.
`spattype=`*seq*+*rand*	Predefined pattern options used for RSC serial test.
`stest=`*u_u*+*c_c*+*d_d*	Defines port and configuration to use for RSC serial test.
`ttyu_baud=`*ALL* | *specific_baud*	Defines baud rates to be used in testing the RSC's console port.
`rscmodem=`*E/D*	Enables or disables the RSC PCMCIA modem test.

Note – 64-bit tests are located in the `sparcv9` subdirectory: `/opt/SUNWvts/bin/sparcv9/`*testname*. 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 "32-Bit and 64-Bit Tests" on page 3.

CHAPTER 46

Serial Asynchronous Interface (PCI) Test (saiptest)

saiptest checks the functionality of the serial asynchronous interface (SAI) card through its device driver.

Note – You must have Patch ID 109338 installed on the system where you plan to run the saiptest.

---

saiptest Hardware Requirements

Before running the SunVTS diagnostics software, make sure you install the device driver and the cards to be tested. Also, you should reboot your system with the boot -r command to reconfigure the system and allow the SunVTS kernel to recognize the new driver.

Note – You must run the saiptest in Intervention mode.

Note – You must have Patch ID 109338 installed on the system where you plan to run the sapitest.

The following minimum hardware configuration is required to successfully run the Internal test:

- PCI-based SPARC system with a PCI slot
- Serial asynchronous interface card, installed in one of the PCI slots

The following hardware is also required to run other SunVTS Serial Asynchronous Interface tests:

- Serial asynchronous interface patch panel (part no. 370-2810)
- 25-pin serial loopback plugs (part no. 540-1558)
- RS-232 serial cables (part no. 530-1685)
- TTY terminal

`saiptest` Options

To reach the 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 may not include the device appropriate to this test. Refer to the *SunVTS User's Guide* for more details.

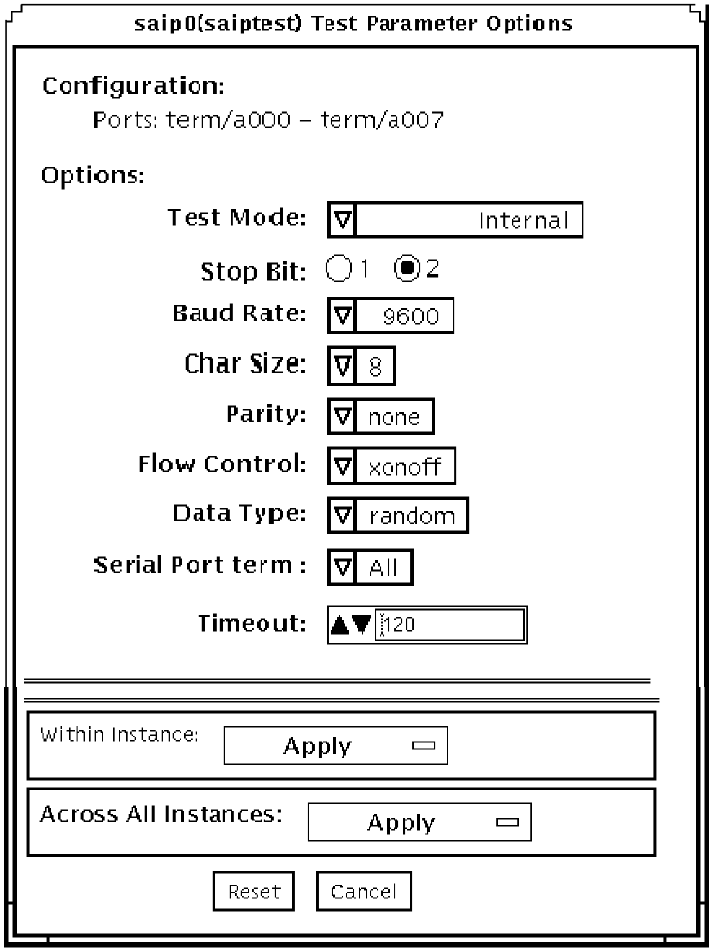


FIGURE 46-1 `saiptest` Test Parameter Options Dialog Box

The Configuration section of the Options dialog box displays the asynchronous serial ports available for the SAI board. Table 4-1 shows the available ports.

TABLE 46-1 `saiptest` Asynchronous Serial Ports

Board Number	Board Device	Serial Ports
0	saip0	term / a000-a007
1	saip1	term / b000-b007
2	saip2	term / c000-c007
3	saip3	term / d000-d007

TABLE 46-1 `saiptest` Asynchronous Serial Ports *(Continued)*

Board Number	Board Device	Serial Ports
4	saip4	term / e000-e007
5	saip5	term / f000-f007
6	saip6	term / g000-g007
7	saip7	term / h000-h007
8	saip8	term / i000-i007
9	saip9	term / j000-j007
10	saip10	term / k000-k007
11	saip11	term / l000-l007
12	saip12	term / m000-m007
13	saip 13	term / n000-n007
14	saip 14	term / o000-o007
15	saip 15	term / p00-p007

TABLE 46-2 `saiptest` Options

`saiptest` **Option**	**Description**
Internal Test	Performs internal loopback testing on the SAI card(s) installed in PCI slots. You do not need to attach anything to the card(s) to perform this test.
25-pin Loopback	Provides full-duplex transmission and full-modem loopback testing of the serial port selected in the Serial Port section of the option menu. Attach the 25-pin loopback plug to the serial port on the serial asynchronous interface patch panel that is being tested. This test cannot be run concurrently with the Echo-TTY option enabled.
Echo-TTY	Checks the proper operation of the serial port selected in the Serial Port selection of the option menu by echoing characters typed on a TTY terminal keyboard to the TTY terminal screen. The characters you type on your TTY keyboard display on the TTY screen. Note: A TTY connection to the serial asynchronous interface serial port requires corresponding character size se up. For example, if a TTY attachment is running with 8-bit character size, then the Char Size `saiptest` option should be set to 8- bits. If you do not type any characters within two minutes, this test times-out.

TABLE 46-2 `saiptest` Options *(Continued)*

`saiptest` **Option**	**Description**
Baud Rate	Specifies the baud rate. Choose 110, 300, 600, 1200, 2400, 4800, 9600, 19200, or 38400 baud. Note: The baud rate of 38400 can only be used if the Internal test is disabled and you are testing one port at a time.
Char Size	Specifies the character length. Choose 5, 6, 7, or 8 characters.
Stop Bit	Specifies the number of stop bits. Choose 1 or 2 bits.
Parity	Specifies the selectable parity. Choose none, odd, or even.
Flow Control	Specifies the selectable flow control. Choose XOnOff, rtscts, or both.
Data Type	Specifies the selectable data type pattern. Choose 0x55555555 (0x55), 0xaaaaaaaa (0xaa), or random.
Serial Port	Specifies the serial port to be tested. The available ports are listed in the Configurations section at the top of the `saiptest` options menu.
Timeout	Specifies the number of seconds until the test times out. The default is 120 seconds.

`saiptest` Test Modes

TABLE 46-3 `saiptest` Test Modes

Test Mode	**Supported?**	**Description**
Connection	No	Not supported.
Functional (Offline)	Yes	Runs the full set of tests.

`saiptest` Command-Line Syntax

`/opt/SUNWvts/bin/saiptest` *standard_arguments* **`-o dev=`***device_name***`,`**
`M=`*test_mode***`,B=baud_rate,Size=`***character_size***`,Stop=`***#of_stop_bits***`,`**
`Parity=`*parity***`,F=`***flow_control***`,Data=`***test_pattern***`,sp=`***serial_port***`,tout=`***time_out*

TABLE 46-4 `saiptest` Command-Line Syntax

Argument	Description
dev=*device_name*	Specifies the asynchronous serial ports of the PCI card slots tested. Since there is no default, you must type a device name—either a board(saip0-16) or an individual port (term / x000-term / x007, where x is a-p): •saip0 = the 8 asynchronous serial ports in the first card •saip1 = the 8 asynchronous serial ports in the second card •saip2 = the 8 asynchronous serial ports in the third card •saip3 = the 8 asynchronous serial ports in the fourth card •saip4 = the 8 asynchronous serial ports in the fifth card •saip5 = the 8 asynchronous serial ports in the sixth card •saip6 = the 8 asynchronous serial ports in the seventh card •saip7 = the 8 asynchronous serial ports in the eighth card •saip8 = the 8 asynchronous serial ports in the ninth card •saip9 = the 8 asynchronous serial ports in the tenth card •saip10 = the 8 asynchronous serial ports in the eleventh card •saip11 = the 8 asynchronous serial ports in the twelfth card •saip12 = the 8 asynchronous serial ports in the thirteenth card •saip13 = the 8 asynchronous serial ports in the fourteenth card •saip14 = the 8 asynchronous serial ports in the fifteenth card •saip15 = the 8 asynchronous serial ports in the sixteenth card or `/dev/term/x00`*m* Where *x* is a-p and m is 0 to 7 (any of the asynchronous serial ports in PCI card slots).
M=*test_mode*	Specifies Internal, 25_pin_loopback, or Echo_TTY test mode.
B=*baud_rate*	Sets the baud rate to 110, 300, 600, 1200, 2400, 4800, 9600, 19200, 38400; the default is 9600.
Stop=*#of_stop_bits*	Toggles the number of stop bits between 1 or 2. The default is 1.
Size=*character_size*	Sets the character size as an integer between 5 and 8.
Parity=*parity*	Specifies the parity as none, odd, or even. The default is none.
F=*flow_control*	Specifies flow control as xonoff, rtscts, or both.
Data=*test_pattern*	Specifies test pattern as 0x55555555, 0xAAAAAAAA, or random.
sp=*serial_port*	Specifies the terminal and asynchronous serial port number, such as term / a00*n* (sp=*n*).
tout=*time_out*	Specifies the number of seconds until the test times out. The default is 120 seconds.

Note – 64-bit tests are located in the `sparcv9` subdirectory: `/opt/SUNWvts/bin/sparcv9/`*testname*. 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 "32-Bit and 64-Bit Tests" on page 3.

Note – The `saiptest` error messages are generated when the SunVTS Serial Asynchronous Interface discovers errors. The error descriptions that appear in the VTS interface identify probable causes for the card or test failure, and identify the Field Replaceable Unit (FRU) and recommended action, if possible. The two FRUs under test are: the Serial Asynchronous Interface card, and the patch panel with the cable attached.

CHAPTER 47

Sun Enterprise Cluster 2.0 Network Hardware Test (scitest)

scitest verifies the functionality of the Sun Enterprise Cluster 2.0 by checking the networking hardware. For this test to be meaningful, the cluster must already be configured before the test is run. For details on how to configure the cluster, refer to *Sun Enterprise Cluster 2.0 Hardware Site Preparation, Planning, and Installation Guide*.

scitest reads the /etc/sma.ip file to determine the target nodes in the cluster. scitest mainly uses the Internet Control Message Protocol (ICMP) to test the connections between cluster nodes.

After finding the cluster nodes (targets), scitest 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.
- Pattern test—sends 256 packets of maximum length, where each packet contains one test pattern and all byte patterns (0 to 0xFF) are used.

Note – scitest is a scalable test. The maximum number of instances is two per SCI card.

---

scitest Options

To reach the 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 may not include the device appropriate to this test. Refer to the *SunVTS User's Guide* for more details.

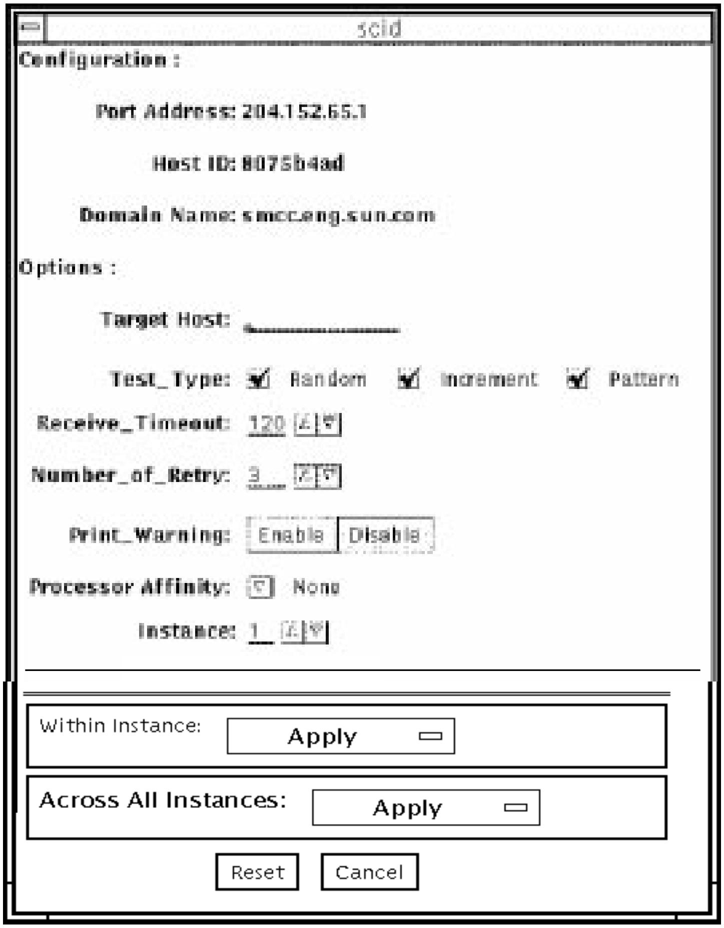


FIGURE 47-1 `scitest` Test Parameter Options Dialog Box

TABLE 47-1 `scitest` Options

`scitest` **Options**	**Description**
Target Host	Not used.
Receive Timeout field	Specify a number between 0–600 seconds. The default is 120 seconds.
Number of Retries field	The number of retries before an error is flagged. Specify a number between 0–128.
Print Warning	Disabled by default. Choose Enable to see warning errors, such as `retry on timeout` errors.

`scitest` Test Modes

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

TABLE 47-2 `scitest` Test Modes

Test Mode	Supported?	Description
Connection	Yes	`scitest` checks if the device is connected. It searches through all the network interfaces for the specified device name. `scitest` finds the device not connected the test fails, otherwise, it returns `device is connected`.
Functional (Offline	Yes	`scitest` performs all three tests (Random test, Incremental test, and Pattern test) sequentially. It allows you to specify an option in such a way that `scitest` performs a very stressful test.

`scitest` Command-Line Syntax

`/opt/SUNWvts/bin/scitest` *standard_arguments* **`-o`**
`dev=`*interface***`,test=`***type***`,packets=`***n***`,pattern=`***hex***`,delay=`***seconds***`,`**
`timeout=`*seconds***`,retry=`***n***`,warn`**

TABLE 47-3 `scitest` Command-Line Syntax

Argument	Description
`dev=`*interface*	Network interface name. The default value is `le0` for Ethernet networks.
`test=`*type*	The test type. Specify `random`, `increment`, or `pattern`. The default value is random+increment+pattern to run.
`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`.
`delay=`*seconds*	Indicates the time between subtests in seconds. The default is 30 seconds.

TABLE 47-3 `scitest` Command-Line Syntax

Argument	Description
`timeout=`*seconds*	Indicates the number of seconds to wait before a timeout. The default is 20 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*. 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 “32-Bit and 64-Bit Tests” on page 3.

CHAPTER **48**

Smart Card Test (`sctest`)

The Smart Card test (`sctest`) consists of a series of subtests to exercise smart cards and smart card readers. It exercises the Java™ card terminal reader driver, the Solaris I2C SCM device driver, and the serial device driver.

`sctest` tests only the internal SC reader. `sctest` is only run as an intervention test. Users need to initalize SC card and Internal SC reader to run `sctest`.

Note – Users should initialize the smart card with commands or GUI provided by OS. This initialization information can be found in the Smart Card User Administration Guide at docs.sun.com.

Caution – Do not use a smart card with real data for `sctest`. Allocate a scratch smart card for testing purposes only. Functions of `sctest` will delete any existing data on the card.

Note – Do not run `sctest` and `sptest` at the same time. Test results will conflict if `sptest` is enabled when `sctest` is run. (Because it is an intervention test, `sptest` is disabled by default.)

To select `sctest`, click the Intervention button in the Select Devices panel.

`sctest` Test Requirements

The following four requirements must be met before running `sctest`:

- You must enable the SunVTS intervention mode.

- You must have a scratch smart card ready.
- You must configure the card reader, if it has not ben configured already. The card reader can be configured with the GUI provided by the OS (under Workspace Menu-Tools-Smart card).
- You must initialize the test card with GUI provided by the OS (under Workspace Menu-Tools-Smart card).

Once you have fulfilled these four requirements, you are ready to run `sctest`.

`sctest` Subtests

TABLE 48-1 `sctest` Subtests

Subtest	Description
Card Presence Detect test	Checks whether a card is present in the reader. Place the card in the reader to begin the test.
Card Lock/Unlock test	Exercises the functions that lock and unlock a card. Place the card in the reader to begin the test.
Insertion/Removal test	Tests detection of card insertion and removal. If the card is already in the reader, remove and reinsert when prompted. This test is disabled by default, since it requires user intervention.
PinCardService test	Waits for a card to be inserted, then performs authentication of the default PIN (`$$$$java`). Note: this subtest assumes the card has been initialized.
Simple Authentication test	Waits for a card to be inserted, then performs authentication of the default PIN (`$$$$java`). Displays the username and password of an example user called AAA. This test only works for cards that have been initialized. This subtest requires the PinCardService test to first validate the PIN on the smart card.
UserInfoCardService test	Waits for a card to be inserted, then performs authentication of the default PIN (`$$$$java`). Displays the user information such as the username and password of example user called AAA. If Write mode is enabled, the test will also create default user information for an example user BBB. If Verbose mode is enabled, this user information will be displayed. Next, the test deletes the user information for BBB. When the test queries for user information for BBB, it expects an exception (since this information is already deleted). This subtest requires that the PinCardService test has validated the PIN on the smart card.

`sctest` Options

To reach the 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 may not include the device appropriate to this test. Refer to the *SunVTS User's Guide* for more details.

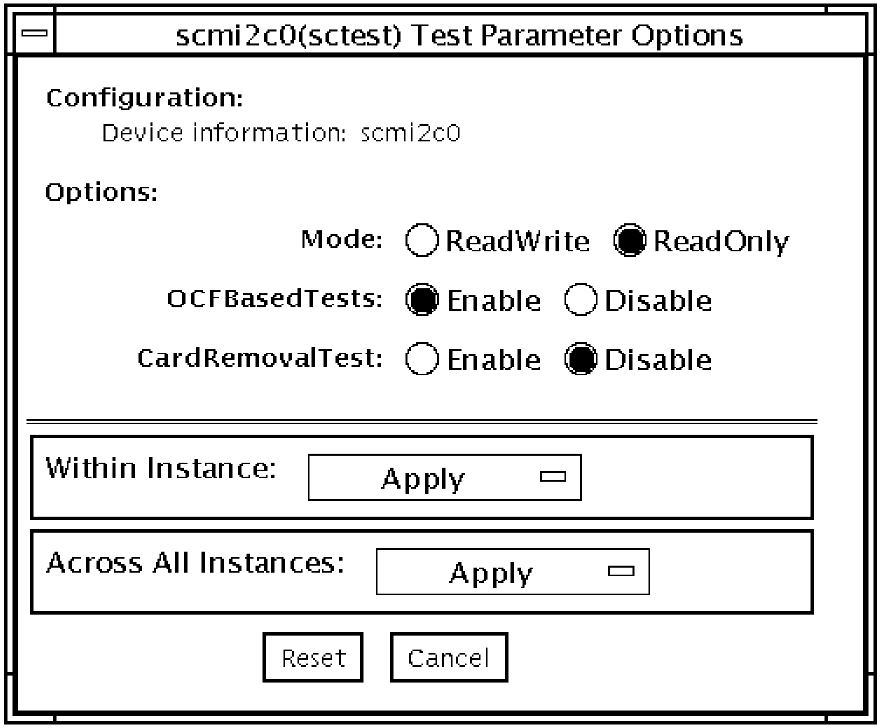


FIGURE 48-1 `sctest` Test Parameter Options Dialog Box

TABLE 48-2 `sctest` Options

`sctest` **Options**	**Description**
Mode	Enables or disables the Write mode for the UserInfoCard Service subtest.
OCF Based tests	Enables or disables all of the following subtests: Card Presence Detect, Card Lock/Unlock, Insertion/Removal, PinCardService, Simple Authentication, and UserInfoCard.
Card Removal test	Enables or disables the Insertion/Removal subtest.

sctest Test Modes

TABLE 48-3 `sctest` Test Modes

Test Mode	Supported?	Description
Connection	No	Not supported.
Functional (Offline)	Yes	Runs the full set of tests.

sctest Command-Line Syntax

/opt/SUNWvts/bin/sctest *standard_arguments*
-o dev=*a | b | i2cscmN***,Mode**=*ReadWrite | ReadOnly*,**OCFBasedTests**=*Enable | Disable*,
CardRemovalTest=*Enable | Disable*

TABLE 48-4 `sctest` Command-Line Syntax

Argument	Description
dev=*a | b | i2cscmN*	Names the type of device. The value **a** or **b** implies `ttya` or `ttyb`. **N**=0, 1, 2, and so on.
Mode=*Enable | Disable*	Enables or disables the Write mode for the UserInfoCard Service subtest.
OCFBasedTests= *Enable | Disable*	Enables or disables all of the following subtests: Card Presence Detect, Card Lock / Unlock, Insertion / Removal, PinCardService, Simple Authentication, and UserInfoCard.
CardRemovalTest= *Enable | Disable*	Enables or disables the Insertion / Removal subtest.

Note – 64-bit tests are located in the `sparcv9` subdirectory: `/opt/SUNWvts/bin/sparcv9/`*testname*. 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 “32-Bit and 64-Bit Tests” on page 3.

CHAPTER **49**

Environmental Sensing Card Test (sentest)

sentest checks the SCSI Environmental Sensing card (SEN) installed in the SPARCstorage RSM to monitor the enclosure environment. The SEN card monitors the enclosure's over-temperature condition, fan-failures, power-supply failures, and drive activity.

sentest verifies the SEN card by first setting each control function to a predetermined value, and then reading it back to verify if the value is correct.

sentest verifies the following control functions in the enclosure:

- Alarm (enable / disable)—sentest toggles the alarm to the disable state, then to the enable state.
- Alarm time (0-0xff seconds)—sentest sets the time (from 0 to 4095), then reads it back to verify the time setting.
- Drive fault LED (DL0-DL6)—sentest toggles each LED to its OFF and ON states.

sentest is a nonscalable test. It cannot verify the control function settings correctly if another instance is changing the setting.

sentest Options

To reach the 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 may not include the device appropriate to this test. Refer to the *SunVTS User's Guide* for more details.

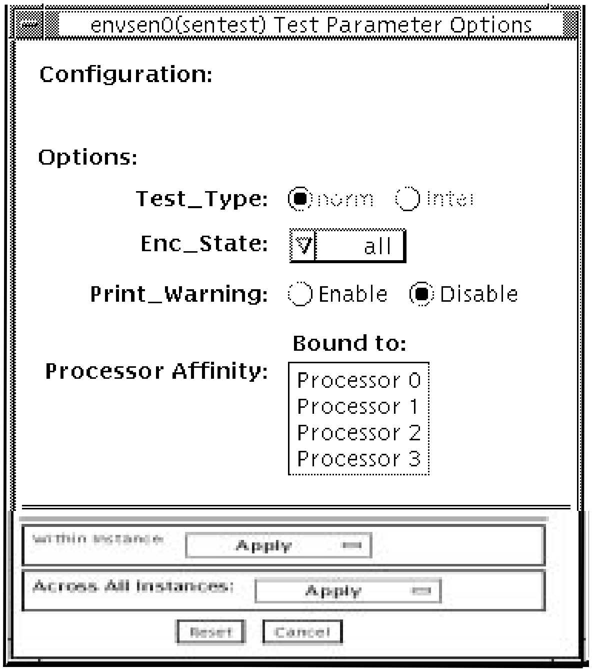


FIGURE 49-1 `sentest` Test Parameter Options Dialog Box

TABLE 49-1 `sentest` Options

`sentest` **Option**	**Description**
Test Type	Specifies the test to be performed. `Norm` test type performs normal testing as alarm enable/disable, alarm time setting, and drive LED on/off testing. Interactive test type reports the current enclosure status.
Enc_state	Specifies which subsystem's status in the enclosure is reported. Default is ALL. This test is only used with the inter test type and in Functional test. The test options are: •Alarm enable/disable status •Drive present status •Drive LED status •Power modules status •Fan modules status •Over temperature, abs (abnormal, no immediate attention needed), chk (abnormal, immediate attention needed) status •All of the above

`sentest` Test Modes

`sentest` supports all three test modes. Each mode performs a different test scheme on the SEN card.

TABLE 49-2 `sentest` Test Modes

Test Mode	Supported?	Description
Connection	Yes	Checks the device connection by opening the device. If the device does not open, the device is not connected.
Functional (Offline)	Yes	Checks three components within the enclosure. It checks alarm enable/disable, alarm time setting, and the drive LEDs. It does not test the power on/off function (only functions whose values can be changed are tested).

`sentest` Command-Line Syntax

`/opt/SUNWvts/bin/sentest` *standard arguments*
`-o dev=`*interface***`,test=`***type***`,enc=`***component*

TABLE 49-3 `sentest` Command-Line Syntax

Argument	Description
`dev=`*interface*	SEN card device name. The default value is `ses0.`
`test=`*type*	Specifies the test type. Select `Norm` for normal testing or `Inter` for interactive testing; the default value is `Norm.` Possible values are: `norm and inter.`
`enc=`*component*	Indicates which part of the enclosure status is reported. The default value is `ALL`. Possible values are: `enalm`, `dp`, `dl`, `pm`, `fan`, `ovt`, and `ALL.`

Note – 64-bit tests are located in the `sparcv9` subdirectory: `/opt/SUNWvts/bin/sparcv9/`*testname*. 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 “32-Bit and 64-Bit Tests” on page 3.

CHAPTER 50

Soc+ Host Adapter Card Test (socaltest)

socaltest aids the validation and fault isolation of the SOC+ host adapter card. In the case of a faulty card, the test tries to isolate the fault to the card, the GBIC module, or the DMA between the host adapter card and the host memory.

Note – Do not run socaltest and enatest at the same time, otherwise test failures might occur.

Note – Do not run socaltest and disktest at the same time, otherwise test failures might occur.

Note – Do not run socaltest with a high system load. Running this test with a large number of instances and concurrency might cause resource limitations that cause this test to fail.

socaltest Options

To reach the 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 may not include the device appropriate to this test. Refer to the *SunVTS User's Guide* for more details.

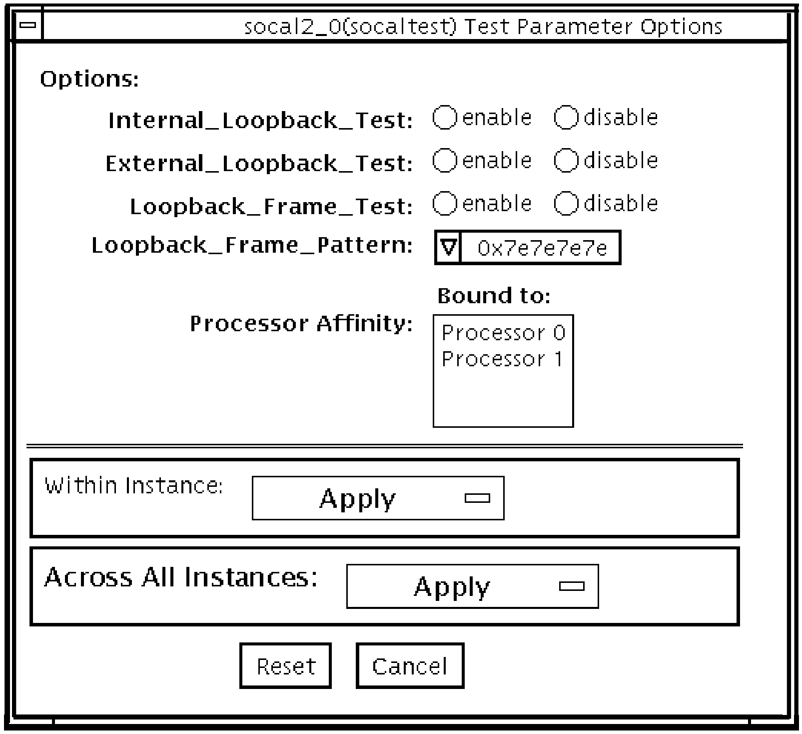


FIGURE 50-1 `socaltest` Test Parameter Options Dialog Box

TABLE 50-1 `socaltest` Options

`socaltest` **Options**	**Description**
Internal Loopback test	Checks the host adapter card and the direct memory access (DMA) with the host system. This is accomplished as follows: 1. A frame is created in the host adapter local memory, sent out through the SOC+ transmitter and internally looped back to the SOC+ receiver. The received data is compared with the original data. 2. A frame is created in the host adapter local memory, sent out through the SOC+ transmitter and looped back through the SERDES (serialiser-deserialiser) chip on the host adapter card. The received data is compared with the original data. 3. A frame is created in the host main memory, transferred through the DMA to the host adapter transmitter, looped back within the SOC+ chip, and transferred from the receiver to the host main memory through the DMA. The received frame is compared with the original transmitted frame, which tests the host memory to the host adapter DMA path.
External Loopback test	The External Loopback test verifies the proper functioning of the GBIC module. A frame is created in the host adapter local memory, sent out and looped back through the external loopback connector attached to the port. If the external loopback test is run together with the internal loopback test, the DMA path is also tested by creating a frame in host main memory, transferring it to the host adapter through the DMA, looping it back through the external loopback connector and transferring the received frame back to the host main memory by DMA.
Loopback Frame test	Sends out a buffer initialized with the selected pattern and compares it with the looped back frame. It passes if the two compare and fails if they do not.
Loopback Frame Pattern	List of selectable patterns for the Loopback Frame test.

Note – In addition to the tests described above, `socaltest` also tests the basic functions of the SOC+ chip, the on-board XRAM, and the host control buffer by invoking the appropriate tests implemented in firmware.

`socaltest` Test Modes

TABLE 50-2 `socaltest` Test Modes

Test Mode	Supported?	Description
Connection	No	Not supported
Functional (Offline)	Yes	Runs the full set of tests

Note – You cannot run the Internal and External Loopback tests if the port is connected to a disk array.

`socaltest` Command-Line Syntax

`/opt/SUNWvts/bin/socaltest` *standard arguments*
`-o dev=`*device name***`,elb=`***enabled | disabled***`,ilb=`***enabled | disabled***`,lbf=`***enable | disable***`,ptn=`***pattern*

TABLE 50-3 `socaltest` Command-Line Syntax

Argument	Description
`dev=`*device name*	The name of the socal port to be tested.
`elb=`*enabled | disabled*	Enables or disables the External Loopback test.
`ilb=`*enabled | disabled*	Enables or disables the Internal Loopback test.
`lbf=`*enable | disable*	Enables or disables the Loopback Frame test.
`ptn=`*pattern*	Specify the pattern in hexadecimal, for example: `ptn=0x7e7e7e7e`

Note – 64-bit tests are located in the `sparcv9` subdirectory: `/opt/SUNWvts/bin/sparcv9/`*testname*. 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 "32-Bit and 64-Bit Tests" on page 3.

CHAPTER **51**

Serial Parallel Controller Test (spiftest)

The spiftest accesses card components such as the cd-180 and ppc2 chips, and the serial and parallel ports through the serial parallel controller device driver.

spiftest Hardware Requirements

Before running the SunVTS system exerciser, make sure you install the cards to be tested and the device driver. You should also reboot your system with the boot -r command to reconfigure the system and allow the SunVTS kernel to recognize the new driver.

Note – The spiftest must be run in Intervention mode.

The following minimum hardware configuration is required to successfully run the Internal test:

- SBus-based SPARC desktop system with an SBus slot
- Serial parallel controller card, installed in one of the SBus slots

The following hardware is also required to run the other SunVTS serial parallel controller tests:

- Serial parallel controller patch panel (part number 540-2007)
- 96-pin loopback plugs (part number 370-1366)
- 25-pin serial loopback plugs (part number 540-1558)
- RS-232 serial cables (part number 530-1685)
- TTY terminal

spiftest Options

To reach the 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 may not include the device appropriate to this test. Refer to the *SunVTS User's Guide* for more details.

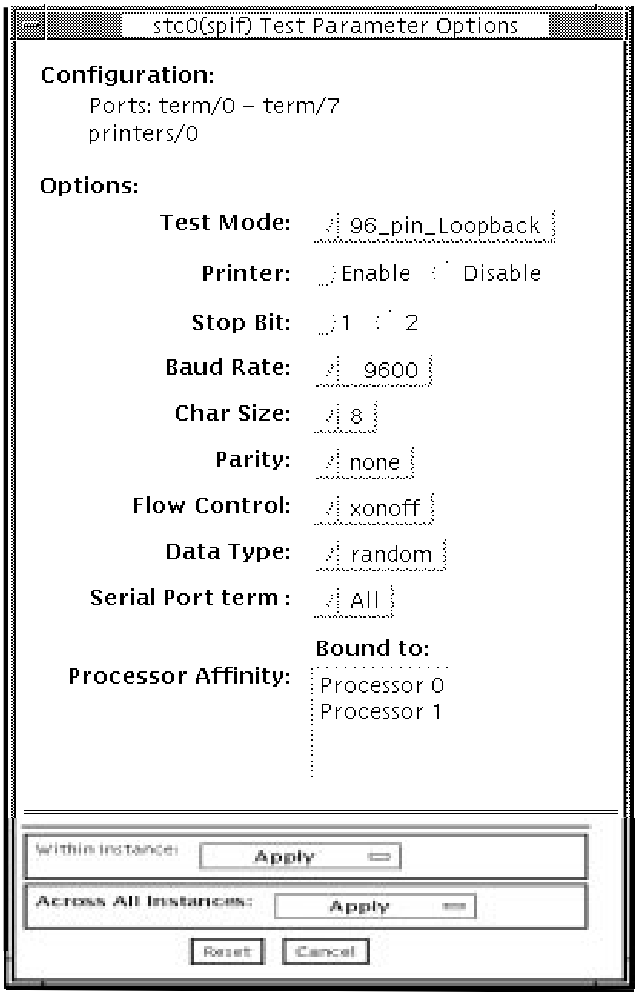


FIGURE 51-1 spiftest Test Parameter Options Dialog Box

The Configuration section of the Options dialog box displays the serial ports available for the SPC/S board. The available ports are listed in the table below.

TABLE 51-1 `spiftest` Serial Ports for the SPC/S Board

Board Number	Board Device	Serial Ports	Parallel Ports
0	stc0	term/0-7	printers/0
1	stc1	term/8-15	printers/1
2	stc2	term/16-23	printers/2
3	stc3	term/24-31	printers/3
4	stc4	term/32-39	printers/4
5	stc5	term/40-47	printers/5
6	stc6	term/48-55	printers/6
7	stc7	term/56-63	printers/7

The `spiftest` options are described in the table below.

TABLE 51-2 `spiftest` Options

spiftest Option	Description
96-pin Loopback (LB)	Provides data transmission, full-modem loopback, and parallel port loopback testing. You *must* attach a 96-pin loopback connector (part number 370-1366) to the card before running this test (see Appendix A, “96-Pin Female Loopback Connector” section).
Internal test	Performs a quick internal check of the serial parallel controller card(s) installed in SBus slots. You do not need to attach anything to the card(s) to perform this test.
25-pin Loopback (LB)	Provides full-duplex transmission and full-modem loopback testing of the serial port selected in the Serial Port selection of this menu. You *must* attach the 25-pin Loopback plug to the serial port on the Serial Parallel Controller Patch Panel that is being tested (see Appendix A). This test cannot be run concurrently with the Echo-TTY option enabled.

TABLE 51-2 `spiftest` Options

spiftest Option	Description
Echo-TTY	Checks the proper operation of the serial port selected in the Serial Port section of this menu by echoing characters typed on a TTY terminal keyboard to the TTY terminal screen. The characters you type should be displayed on the TTY screen. Note: TTY connection to the `spif` serial port requires corresponding character size setup. For example, if a TTY attachment is running with 8-bit character size, then the `spiftest` option 'Char Size' should also be set to 8 bits. If you do not type within two minutes, this test will time-out. Terminate testing by pressing Control-C. After a short delay, the Status window updates the Pass Count. This test cannot be run concurrently with the 25-pin Loopback subtest.
Printer	Sends the entire ASCII character set to a parallel printer. You must attach a parallel printer to the parallel port on the Serial Parallel Controller patch panel. Observe the printer output to validate the test.
Baud Rate	Specifies the baud rate; choose 110, 300, 600, 1200, 2400, 4800, 9600, 19200, or 38400 baud. Note: The baud rate of 38400 can only be used if one port is tested at a time and the Internal Test is disabled.
Char Size	Indicates character length. Choose 5, 6, 7, or 8 characters.
Stop Bit	Specifies the number of stop bits. Choose 1 or 2 bits.
Parity	Specifies the selectable parity. Choose `none`, `odd` or `even`.
Flow Control	Specifies the selectable flow control. Choices are `xonoff`, `rtscts`, or `both`.
Data Type	Specifies the selectable data type pattern. It can be 0x55555555 (0x55), 0xaaaaaaaa (0xaa), or random.
Serial Port	Specifies the serial port to be tested. The available ports are listed in the Configurations section at the top of the `spiftest` Options dialog box.

You can also change the test options by modifying the `/opt/SUNWvts/bin/.customtest` file. See "Adding Your Own Tests" in `.customtest` in the *SunVTS User's Guide.*

`spiftest` Test Modes

TABLE 51-3 `spiftest` Test Modes

Test Mode	Supported?	Description
Connection	No	Not supported
Functional (Offline)	Yes	Runs the full set of tests

`spiftest` Command-Line Syntax

`/opt/SUNWvts/bin/spiftest` *standard_arguments* **`-o dev=`***device_name***`,`** **`M=`***test_mode***`,Ptr=`***enable/disable***`,B=`***baud_rate***`,Size=`***character_size***`,S=`***#of_stop_bits***`,`** **`Parity=`***parity***`,F=`***flow_control***`,Data=`***test_pattern***`,sp=`***serial_port*

TABLE 51-4 `spiftest` Command-Line Syntax

Argument	Description
`dev=`*device_name*	Specifies the serial ports in SBus card slots (0-63) being tested. Since there is no default, you must type a board name: • `stc0`—the 8 serial ports in the first card • `stc1`—the 8 serial ports in the second card • `stc2`—the 8 serial ports in the third card • `stc3`—the 8 serial ports in the fourth card • `stc4`—the 8 serial ports in the fifth card • `stc5`—the 8 serial ports in the sixth card • `stc6`—the 8 serial ports in the seventh card • `stc7`—the 8 serial ports in the eighth card
`M=`*test_mode*	Specifies `Internal, 96_pin_Loopback, 25_pin_loopback,` or `Echo_TTY` test mode.
`Ptr=`*printer_test*	Enables or disables the Printer subtest.
`B`=*baud_rate*	Sets the baud rate to 110, 300, 600, 1200, 2400, 4800, 9600, 19200, or 38400. The default is 9600. To use the 38400 rate, only one port at a time can be tested, and the `Internal` test must be disabled.
`Stop=`*#of_stop_bits*	Toggles the number of stop bits between 1 or 2. The default is 1.
`Size=`*character_size*	Sets character size as a number between 5 and 8.

TABLE 51-4 `spiftest` Command-Line Syntax

Argument	Description
P=*parity*	Specifies the parity as none, odd, or even. The default is none.
F=*flow_control*	Specifies flow control as `xonxoff,rtscts`, or both.
Data=*test_pattern*	Specifies test pattern as 0x55555555, 0xAAAAAAAA, or random.
sp=*serial_port*	Specifies the terminal and serial port number, such as `term/3`.

Note – 64-bit tests are located in the `sparcv9` subdirectory: `/opt/SUNWvts/bin/sparcv9/`*testname*. 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 “32-Bit and 64-Bit Tests” on page 3.

CHAPTER 52

Serial Ports Test (sptest)

sptest checks the system's on-board serial ports (zs[0,1], zsh[0,1], se[0,1], se_hdlc[0,1]), as well as any multi-terminal interface (ALM2) boards (mcp[0-3]). Data is written and read in asynchronous and synchronous modes utilizing various loopback paths.

Intervention mode must be enabled to run this test.

This test is nonscalable.

sptest dynamically probes for se, zs, zsh, and se_hdlc ports for testing. All error messages, warning messages, and options in the test parameter options dialog box are set dynamically to the correct port names. For example, if a system's a and b ports are su ports, and c and d are se ports, sptest will test ports c and d. The test parameter dialog box will also show the correct ports in its menu options, such as "Test Type: c to d," etc. See FIGURE 52-1 on page 312 for an example of the test parameter dialog box.

TABLE 52-1 Serial Port Tests

Mode	Description
Asynchronous testing	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. The user can select the loopback paths to use, the amount of data to transfer, and the baud rate to use. 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.
Synchronous testing	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 sptest 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 then 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.

`sptest` Synchronous Testing Software Requirements

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

▼ To Create Synchronous Devices

1. **Verify that the following two lines are in the `/etc/devlink.tab` file. If they are not there, add them.**

```
type=ddi_pseudo;name=zsh zsh\M0
type=ddi_pseudo;name=clone;minor=zsh zsh
```

Caution – The white spaces in the lines above *must* be a single tab character before and after the `zsh` variables; using spaces will not work.

---

2. **When the lines have been added to the `/etc/devlink.tab` file, change directories to `/kernel/drv` and run the `add_drv zsh` command.**

3. **If this command does not work, run the `rem_drv zsh` command and then run the `add_drv zsh` command again.**

`sptest` Options

To reach the 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 may not include the device appropriate to this test. Refer to the *SunVTS User's Guide* for more details.

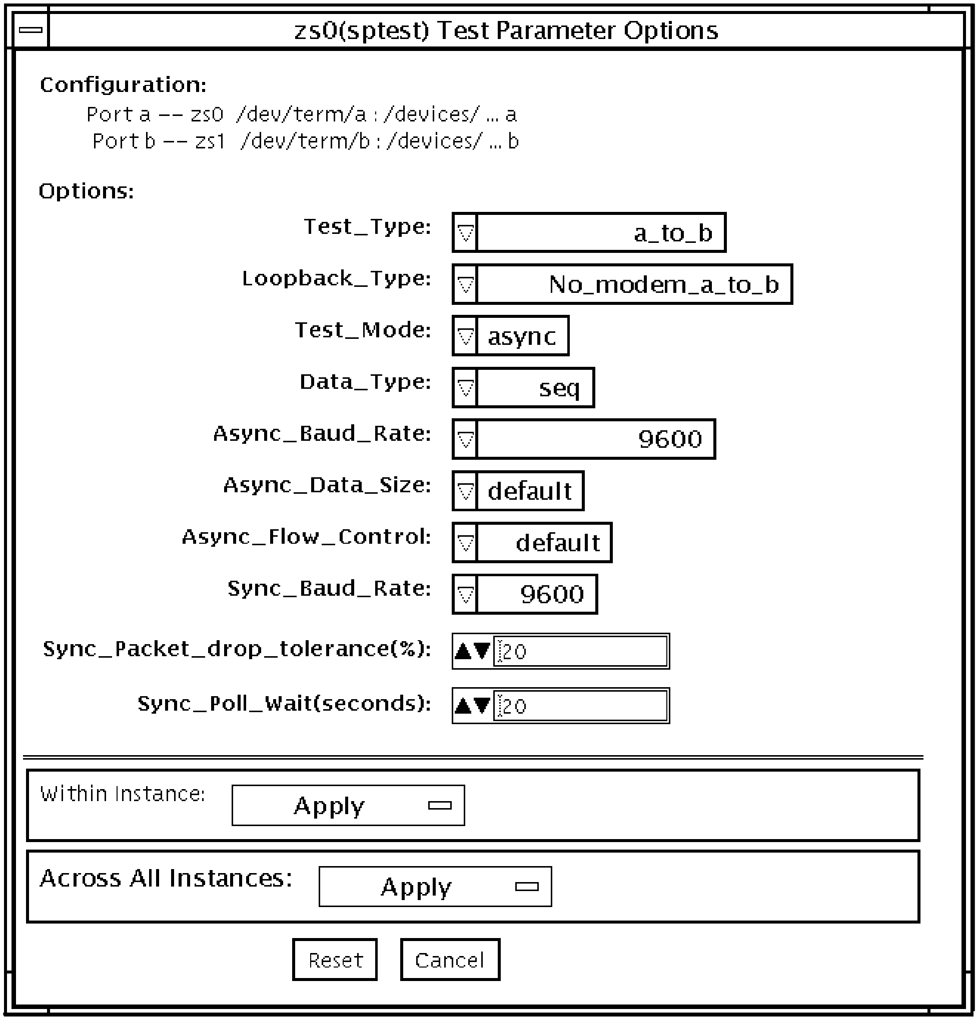


FIGURE 52-1 `sptest` Test Parameter Options Dialog Box

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 external loopback connectors are described in Appendix A. The exact type of loopback connector required depends on the system I/O panel.

Some examples of loopback test commands follow. For a full description of command-line syntax, see “sptest Command-Line Syntax” on page 316.

To test /dev/term/b from the command line using internal loopback, type the following:

```
% ./sptest -vf -o M=async,T=b,L=I
```

To test /dev/term/b from the command line using external loopback, type:

```
% ./sptest -vf -o M=async,T=b,L=P
```

For zs(7D) machines, the internal loopback path is only active in synchronous mode:

```
% ./sptest -vf -o M=sync,T=b,L=I
```

The following table lists the possible devices for each port. Below, "a" represents port a of the CPU board (motherboard), "b" represents port b of the CPU board, and the device names of the ports for each CPU board are listed.

TABLE 52-2 sptest Serial Devices

CPU	Port	Async Device	Sync Device
0	a b	zs0 or se0 zs1 or se1	zsh0 or se_hdlc0 zsh1 or se_hdlc1
1	a b	zs2 zs3	zsh2 * zsh3 *
2	a b	zs4 zs5	zsh4 * zsh5 *

* Currently, only zsh0 and zsh1 are supported by device drivers.

TABLE 52-3 `sptest` Options

sptest Options	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.
Test Mode	Selects the mode to put the serial device into before running the test. The modes available are Asynchronous, Synchronous or Both. When Both is selected, the test runs in Asynchronous mode then Synchronous mode.
Data Type	Selects the data pattern to transfer. The user 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. The test will return an error if you try to use a higher baud rate then is supported. For baud rates greater then 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. The user 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 52-3 `sptest` Options

sptest Options	Description
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 then is supported. For rates greater then 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 Drop Tolerance	Selects the tolerance level of Synchronous mode dropped packets during the many_packets subtest. The default is 20 percent. The valid range is from 0 percent to 99 percent. Some packet drops are expected especially at higher clock 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.

`sptest` Test Modes

`sptest` supports all three SunVTS test modes.

TABLE 52-4 `sptest` Test Modes

Test Mode	Supported?	Description
Connection	Yes	Attempts to open the port to determine if the device is connected. If it fails and 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 (Offline)	Yes	Performs the selected loopback test.

`sptest` Command-Line Syntax

`/opt/SUNWvts/bin/sptest` *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*

TABLE 52-5 `sptest` 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` • ..., ...
`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
`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. The user can select: • `Random` • `Sequential` • `Alphanumeric` • `0x00-0xFF`

TABLE 52-5 `sptest` Command-Line Syntax *(Continued)*

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 then 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; 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`. Note: Some platforms can only support synchronous rates up to 38400 or 76800. For rates greater then 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_waitt*	Synchronous mode additional wait time during poll (in seconds).

Note – 64-bit tests are located in the `sparcv9` subdirectory: `/opt/SUNWvts/bin/sparcv9/`*testname*. 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 “32-Bit and 64-Bit Tests” on page 3.

CHAPTER **53**

SunButtons Test (`sunbuttons`)

The `sunbuttons` test verifies that the SunButtons™ graphics manipulation device is working correctly.

Note – Do not run `sunbuttons` and `sundials` at the same time.

Note – To access this test, you must first install the `SUNWvtsol` package.

The SunVTS OPEN LOOK user interface does not support the latest SunVTS features and will be discontinued when the OPEN LOOK environment is discontinued in the Solaris operating environment. When this occurs, the SunVTS OPEN LOOK tests (sundials and sunbuttons) will also be discontinued. For full feature support, use the SunVTS CDE or TTY interface. Refer to the Solaris "End-of-Software Support Statements" section of the Solaris operating environment release notes for the latest end of support news.

`sunbuttons` Test Requirements

Install SunVTS making sure that the following packages are installed: `SUNWdial` and `SUNWdialh`.

Running the `sunbuttons` Tests

The following tests verify the functionality of SunButtons:

▼ To Use the `.customtest` File

Modify the `.customtest` file as described below to verify that each button functions. Each button lights up in a round-robin fashion.

Note – The `.customtest` file is located in two places. Modify the 32-bit (in the `/opt/SUNWvts/bin` directory) or the 64-bit (in the `/opt/SUNWvts/bin/sparcv9` directory) `.customtest` file according to the Solaris environment that you are using.

---

1. **Edit the `.customtest` file to include the following lines:**

```
sunbuttons;sunbuttons; Mode<CYCLE|diag|diag|mode>
sundials;sundials; Mode<CYCLE|diag|diag|mode>
```

You may have to execute the `xhost +` command.

▼ To Configure the Port Using the `bdconfig` Command

1. **Run the `/usr/sbin/bdconfig` command to configure the `/dev/term/a` or `/dev/term/b` port and enable it.**

   The following example shows how to check a configured serial port b:

```
# ls -l /dev/bd*
lrwxrwxrwx 1 root   other    11 Nov 12 15:39 /dev/bd -> /dev/term/b
#
```

2. **Connect the SunButtons or SunDials device to your configured serial port (a or b).**

▼ To Start SunVTS

After starting SunVTS, you should see Customtest as one of the available options with `sundials` and `sunbuttons` available for selection.

1. **Change to the Functional test mode and enable Intervention.**
2. **Enable the `sunbuttons` and `sundials` tests.**

3. **Start testing.**

   In Diag mode, a pop-up window is displayed.

4. **Select the Diagnostics button.**

   Let the test run its course (the Diagnostics button will be selectable again).

5. **Close the pop-up window.**

   The test should register a single pass.

Note – Do not run the `sundials` and `sunbuttons` tests at the same time.

▼ To Run the `sunbuttons` Test From a Shell Command Line

This is an interactive test. The test displays a screen representation of the buttonbox (see FIGURE 53-1) where you can press each of the buttons and see the corresponding button's display change.

1. **Be sure that the buttonbox is connected to one of the serial ports, and that the buttonbox has a power transformer.**

2. **To run the diagnostic test, select the Diagnostics Button on the top of the menu representation.**

   The buttonbox buttons on the screen do not change while the diagnostics test is running.

   There is no option dialog box for this test.

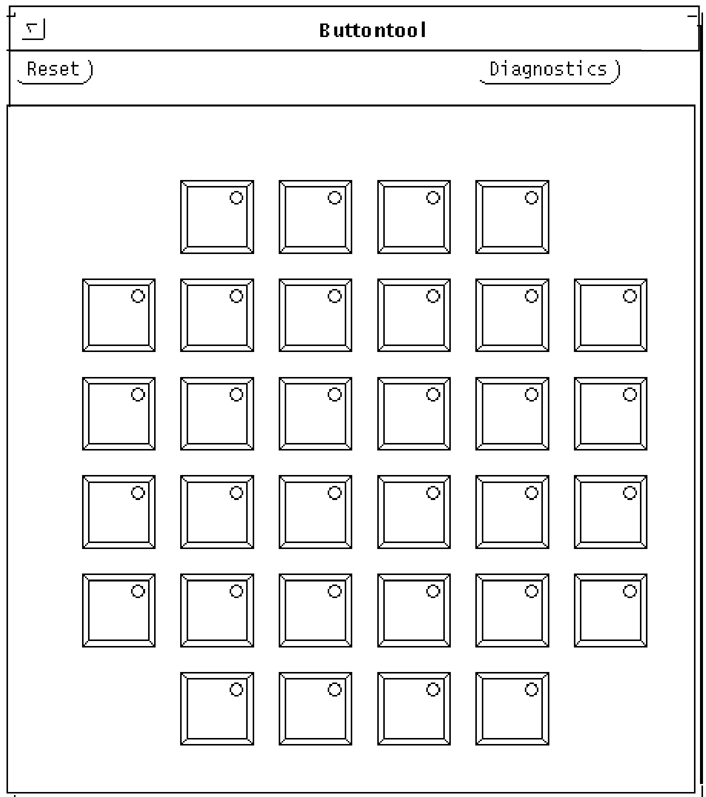


FIGURE 53-1 `sunbuttons` Test Tool

`sunbuttons` Test Modes

TABLE 53-1 `sunbuttonstest` Test Modes

Test Mode	Supported?	Description
Connection	No	Not supported
Functional (Offline)	Yes	This mode verifies the functionality of each button, and lights each button in sequence.

`sunbuttons` Command-Line Syntax

`/opt/SUNWvts/bin/sunbuttons diag` *standard_arguments*

Note – 64-bit tests are located in the `sparcv9` subdirectory: `/opt/SUNWvts/bin/sparcv9/`*testname*. 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 “32-Bit and 64-Bit Tests” on page 3.

CHAPTER 54

SunDials Test (`sundials`)

The `sundials` test verifies that the SunDials™ graphics manipulation device controls are working properly. `sundials` also verifies the connection between the dialbox and serial port.

Note – Do not run `sundials` and `sunbuttons` at the same time.

Note – To access this test, you must first install the `SUNWvtsol` package.

The SunVTS OPEN LOOK user interface does not support the latest SunVTS features and will be discontinued when the OPEN LOOK environment is discontinued in the Solaris operating environment. When this occurs, the SunVTS OPEN LOOK tests (sundials and sunbuttons) will also be discontinued. For full feature support, use the SunVTS CDE or TTY interface. Refer to the Solaris "End-of-Software Support Statements" section of the Solaris operating environment release notes for the latest end of support news.

Running the `sundials` Test

The following tests verify SunDial functionality.

▼ Using the `.customtest` File

1. **Modify the** `.customtest` **file to test the SunDials.**

   Here is an example of a `.customtest` entry for `sundials`:

```
sundials;sundials;text<TEXT|20|-s|>
```

Note – The `.customtest` file is located in two places. Modify the 32-bit (in the `/opt/SUNWvts/bin` directory) or the 64-bit (in the `/opt/SUNWvts/bin/sparcv9` directory) `.customtest` file according to the Solaris environment that you are using.

▼ To Run the `sundials` Test From a Command Line

Running the `sundials` test from a command line starts an interactive test that displays a screen representation of the dialbox (see FIGURE 54-1). You can move each of the dials and see the corresponding dial's display change.

1. **Be certain that the dialbox is connected to one of the serial ports and that the dialbox has a power transformer.**

2. **To run the interactive test, select the Diagnostics button on the top of the window representation. The dialbox has a power transformer.**

   The `sundials` Test dialog box is shown in FIGURE 54-1. There is no Options dialog box for this test.

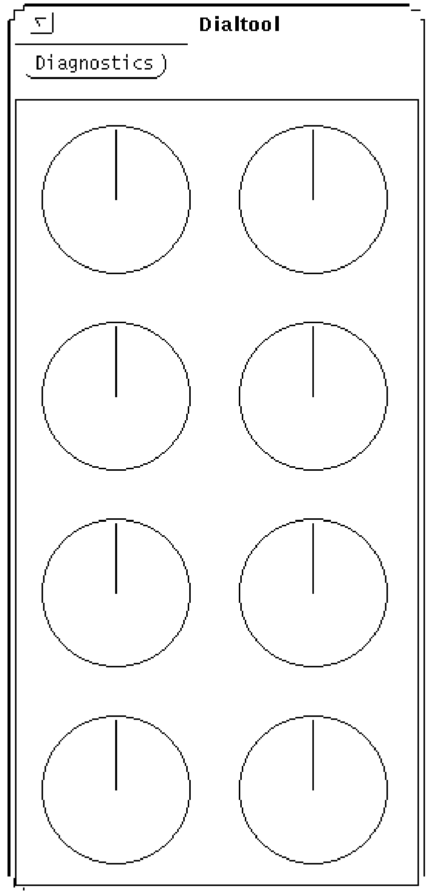


FIGURE 54-1 `sundials` Test Dialog Box

`sundials` Test Modes

TABLE 54-1 `sundials` Test Modes

Test Mode	Supported?	Description
Connection	No	Not supported.
Functional (Offline)	Yes	This mode verifies the connection between the dialbox and serial port. The test also verifies all manipulation device controls.

`sundials` Command-Line Syntax

`/opt/SUNWvts/bin/sundials diag` *standard_arguments*

Note – 64-bit tests are located in the `sparcv9` subdirectory: `/opt/SUNWvts/bin/sparcv9/`*testname*. 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 “32-Bit and 64-Bit Tests” on page 3.

CHAPTER 55

SunHSI Board Test (sunlink)

The sunlink test verifies the functionality of the SBus and PCI bus SunHSI™ boards by using the HDLC protocol. sunlink initializes and configures the selected channel.

Next, sunlink opens a datagram socket and tries to modify the socket to accept ioctl communications with the driver, and receive synchronous mode information from it.

sunlink then opens the ports, linking the upper and lower layers with ioctl calls. After initialization, this test checks for activity before attempting to send or receive data. An error message is returned if activity is detected; otherwise the transmit buffer is filled with random data. Random data is used by default. You may also specify other patterns. The data is then transmitted. If the transmission succeeds, sunlink then receives the returned data and verifies that it is identical to what was sent. Finally, statistics about the send and receive are gathered from the socket.

A full sunlink test takes approximately eight minutes per port and makes a brief check of the board ports before the actual test begins. If the port is bad, the test immediately aborts and returns an error message.

sunlink Test Requirements

This test will not pass unless you install the correct loopback connectors or port to port cables on the ports you are testing. The ports specified for test in the Options dialog box must have loopback connectors attached. See Appendix A for loopback connector part numbers and wiring instructions.

`sunlink` Options

To reach the 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 may not include the device appropriate to this test. Refer to the *SunVTS User's Guide* for more details.

The Configuration field displays the available ports. (See FIGURE 55-1.)

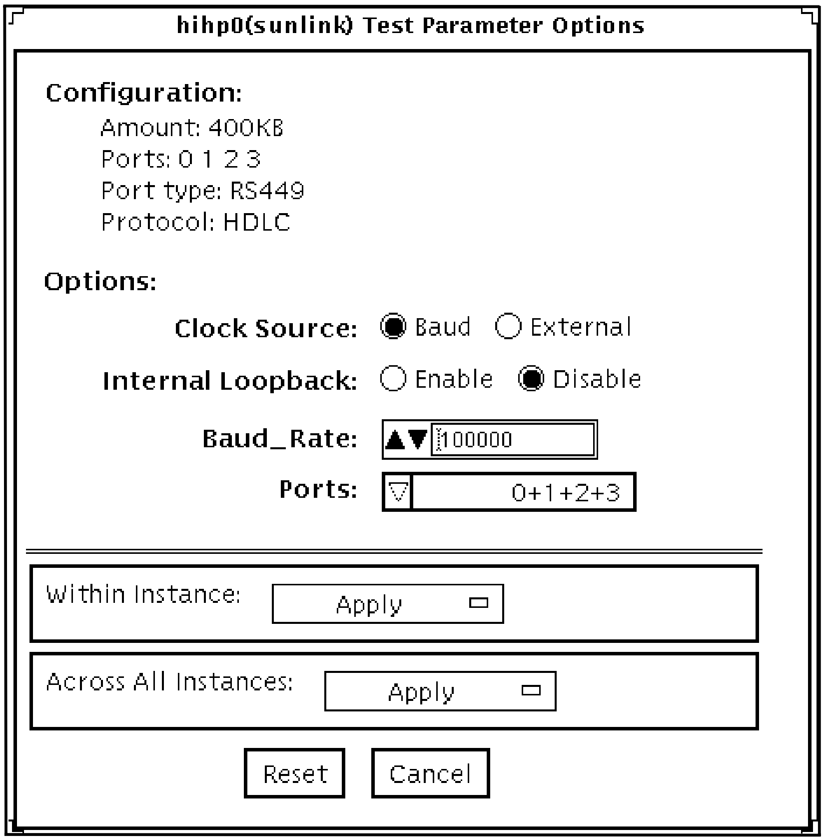


FIGURE 55-1 `sunlink` Test Parameter Options Dialog Box

TABLE 55-1 `sunlink` Options

`sunlink` **Options**	**Description**
Clock source	Select either the onboard clock or an external clock for use when using `sunlink`. To use the external clock option, the transmit, receive, and clock data lines must be physically loopbacked.
Internal Loopback	Enables or disables internal loopback tests. Internal Loopback is only needed when the Loopback setting is not port-to-port, and the clock source is onboard.
Port	Specifies the loopback type—simple single external port loopback, multiple external port loopback, and port-to-port external loopback.

`sunlink` Loopback Connectors

Refer to Appendix A of this manual for information on Sunlink™ loopback cables and loopback connectors. Refer to the High Speed Serial Interface hardware manuals for information on null modem cables.

`sunlink` Test Modes

TABLE 55-2 `sunlinktest` Test Modes

Test Mode	Supported?	Description
Connection	No	Not supported.
Functional (Offline)	Yes	Runs the full set of tests.

`sunlink` Command-Line Syntax

`/opt/SUNWvts/bin/sunlink` *standard_arguments* **`-o dev=`***device_name***`,p=`***port#***`,`** **`P=`***data_pattern***`,I,C=`***clocksource*

TABLE 55-3 `sunlink` Command-Line Syntax

Argument	Explanation
`dev=`*device_name*	Specifies the device to be tested. Use `hih0` for the HDLC protocol
`p=`*ports*	Specifies the port number to be tested.
`P=`*data_pattern*	Specifies the *data_pattern* as one of the following: • `c`—Character (0x55) • `i`—Incrementing • `d`—Decrementing • `r`—Random (default)
`I`	Enables internal loopback for HSI.
`c=`*clocksource*	Specifies the clock source value as one of the following: • `B`—Onboard clock source • `E`—External clock source

The following is a typical command-line syntax for testing a SunHSI board:

```
# /opt/SUNWvts/bin/sunlink -o dev=hih0,P=0+1+2+3
```

This command tests the internal loopback for ports 0, 1, 2, and 3. It does not run for the port to port internal loopback test.

Note – 64-bit tests are located in the `sparcv9` subdirectory: `/opt/SUNWvts/bin/sparcv9/`*testname*. 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 “32-Bit and 64-Bit Tests” on page 3.

CHAPTER 56

Sun PCi Test (sunpcitest)

The sunpcitest tests the SunPCi™ plug-in PCI card, which is an X86 processor embedded in an add-on card. The sunpcitest also monitors and reports messages from the processor self-tests.

sunpcitest Test Requirements

Before running the test, the X-window for the SunPCi must be shut down. Otherwise, the test will not launch.

sunpcitest Options

To reach the 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 may not include the device appropriate to this test. Refer to the *SunVTS User's Guide* for more details.

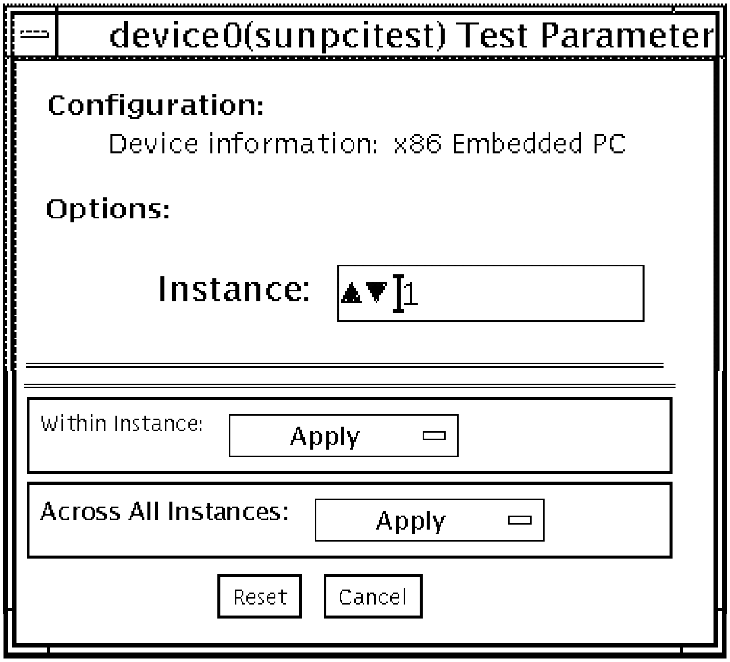


FIGURE 56-1 `sunpcitest` Test Parameter Options Dialog Box

`sunpcitest` only runs with the default parameters in place. Thus, this test does not allow any options to be configured specifically for an individual system. The number of instances is preset to 1 (the default value), as only one local copy of the test is supported.

sunpcitest Test Modes

TABLE 56-1 sunpcitest Test Modes

Test Mode	Supported?	Description
Connection	No	Not supported
Functional (Offline)	Yes	Runs the full set of tests

sunpcitest Command-Line Syntax

/opt/SUNWvts/bin/ftcputest *standard_arguments*

Note – There are no test-specific options for sunpcitest.

Note – 64-bit tests are located in the sparcv9 subdirectory: /opt/SUNWvts/bin/sparcv9/*testname*. 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 "32-Bit and 64-Bit Tests" on page 3.

CHAPTER 57

Sun PCi2 Test (`sunpci2test`)

The `sunpci2test` tests the SunPCi2 plug-in PCI card, which is an Intel platform processor embedded in an add-on card. This test consists of approximately 150 POST routines that perform diagnostic, hardware detection, and initialization functions. The Bridge Diagnostic Tests and System Diagnostic Tests are purely diagnostic in nature. Most other routines are hardware detection and initialization tests, which also can fail.

The `sunpci2test` also monitors and reports messages from the processor self-tests.

`sunpci2test` Test Requirements

Before running the test, the X-window for Microsoft Windows must be shut down. If this is not done, the test will not launch.

▼ To Shut Down Microsoft Windows and SunPCi2 Card:

1. **Click Start button in Microsoft Windows.**
2. **Click Shut Down.**

   The shutdown window appears. Wait for the “It is now safe to shut off your PC” message.
3. **Select “File” from the SunPCi window.**
4. **Select “Exit” from the file menu.**
5. **Click OK.**

sunpci2test Options

To reach the 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 may not include the device appropriate to this test. Refer to the *SunVTS User's Guide* for more details.

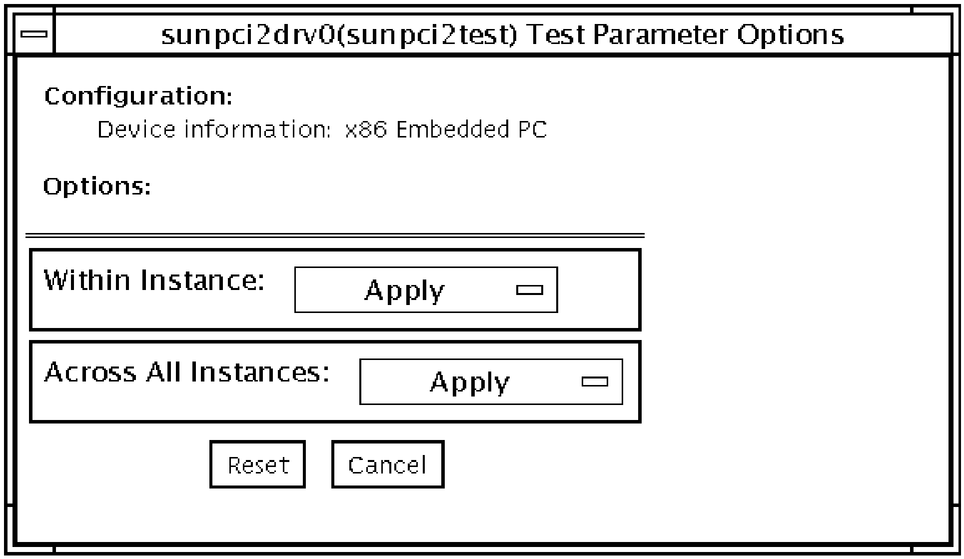


FIGURE 57-1 sunpci2test Test Parameter Options Dialog Box

sunpci2test only runs with the default parameters in place. Thus, this test does not allow any options to be configured specifically for an individual system. The number of instances is preset to 1 (the default value), as only one local copy of the test is supported.

`sunpci2test` Test Modes

TABLE 57-1 `sunpci2test` Test Modes

Test Mode	Supported?	Description
Connection	Yes	Runs the full set of tests
Functional (Offline)	Yes	Runs the full set of tests

`sunpci2test` Command-Line Syntax

`/opt/SUNWvts/bin/sunpci2test` *standard_arguments*

Note – There are no test-specific options for `sunpci2test`.

Note – 64-bit tests are located in the `sparcv9` subdirectory: `/opt/SUNWvts/bin/sparcv9/`*testname*. 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 “32-Bit and 64-Bit Tests” on page 3.

CHAPTER **58**

SuperI/O Test (sutest)

sutest checks the on-board system serial ports (su[0,1]). Data is written and read in asynchronous mode, using various loopback paths. You can select the loopback paths to use, the amount of data to transfer, and the baud rate.

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.

This test uses the asynchronous communication protocol. The termio(7I)interface is used for configuring port characteristics.

For CPU 0, port a on the CPU board (motherboard) uses the su0 asynchronous device, while port b uses su1.

Intervention mode must be enabled to run this test.

This test is nonscalable.

Loopback Connectors

This test requires null modem and plug connectors, which are described in Appendix A “Loopback Connectors”.

There are a variety of loopback paths available. The exact type of loopback connector required depends on the system I/O panel.

The loopback for the “Null Modem a to b” option is a female-to-female plug. Its pin configuration is the same as the one described for the “9-Pin to 9-Pin Port-to-Port Loopback Cable” on page 388.

The loopback for the “Plug a to a” option is described in the section “9-pin Female Single-Port Loopback Plug” on page 386.

Note – `sutest` supports any pair of ports, not just a and b.

`sutest` Options

To reach the 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 may not include the device appropriate to this test. Refer to the *SunVTS User's Guide* for more details.

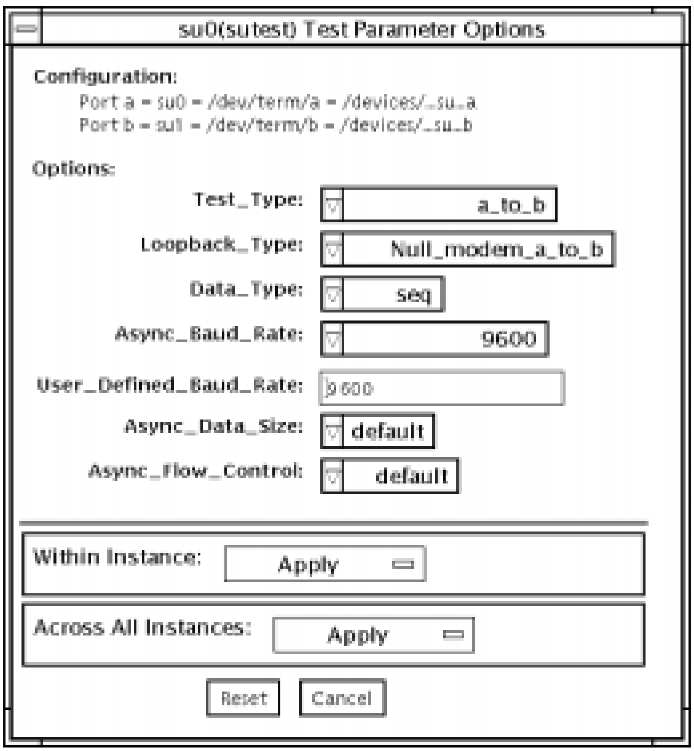


FIGURE 58-1 `sutest` Test Parameter Options Dialog Box

TABLE 58-1 `sutest` Options

sutest Options	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_a_to_a__b_to_b_` 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. •`null_modem_a_to_b` is an external loopback cable for a_to_b and a_to_b_concurrent test types.
Data Type	Selects the data pattern to transfer. The user can select: •`Random` •`Sequential` •`Alphanumeric` •`0x00-0xff`
Async Baud Rate	Selects the baud rate for Asynchronous mode testing. The valid rates are: 50, 110, 300, 600, 1200, 4800, 9600, 19200, 38400, 57600, 115200, and ALL. The default rate is 9600 baud. Some platforms can only support up to 38400 or 76800. The test will return an error if you try to use a higher baud rate then is supported. For baud rates greater then 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.
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 Baud Rate menu. Then enter the new value in the User Defined Baud Rate field.
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. The user 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.

`sutest` Test Modes

`sutest` supports all three SunVTS test modes.

TABLE 58-2 `sutest` Test Modes

Test Mode	Supported?	Description
Connection	Yes	Tries to open the port to determine if the device is connected. If it fails and 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 (Offline)	Yes	Performs the selected loopback test.

`sutest` Command-Line Syntax

`/opt/SUNWvts/bin/sutest` *standard_arguments* **`-o dev=`***device_name*`,` **`porta=`***port_name*`,`**`T=`***test_type*`,`**`L=`***loopback_type*`,`**`D=`***data_pattern*`,` **`AB=`***baud_rate*`,`**`BS=`***write_size*`,`**`F=`***flow_control*

TABLE 58-3 `sutest` Command-Line Syntax

Argument	Explanation
`dev=`*device_name*	Identifies the serial port(s) to test. There is no default value. You must specify a device name such as `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.

TABLE 58-3 `sutest` Command-Line Syntax *(Continued)*

Argument	Explanation *(Continued)*
L=*loopback_type*	The type of loopback connector attached to ports: • Internal_a_to_a__b_to_b • Null_modem_a_to_b • Plug_a_to_a__b_to_b
D=*data_pattern*	Selects the data pattern to transfer. The user can select: • `Random` • `Sequential` • `Alphanumeric` • `0x00-0xFF`
AB=*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 then 153600 the serial line drivers must be set for RS-423 mode and not RS-232 mode.
BS=*write_size*	Asynchronous mode write size; from 1 to 10000 bytes.
F=*flow_control*	Asynchronous mode flow control • `Hardware` (RTS/CTS) • `Software` (xon/xoff) • `None`

Note – 64-bit tests are located in the `sparcv9` subdirectory: `/opt/SUNWvts/bin/sparcv9/`*testname*. 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 “32-Bit and 64-Bit Tests” on page 3.

CHAPTER 59

System Test (systest)

`systest` checks the CPU board by exercising the I/O, memory, and CPU channels simultaneously as threads. There is no quick test option for `systest`; it is a CPU stress test.

systest Options

To reach the 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 may not include the device appropriate to this test. Refer to the *SunVTS User's Guide* for more details.

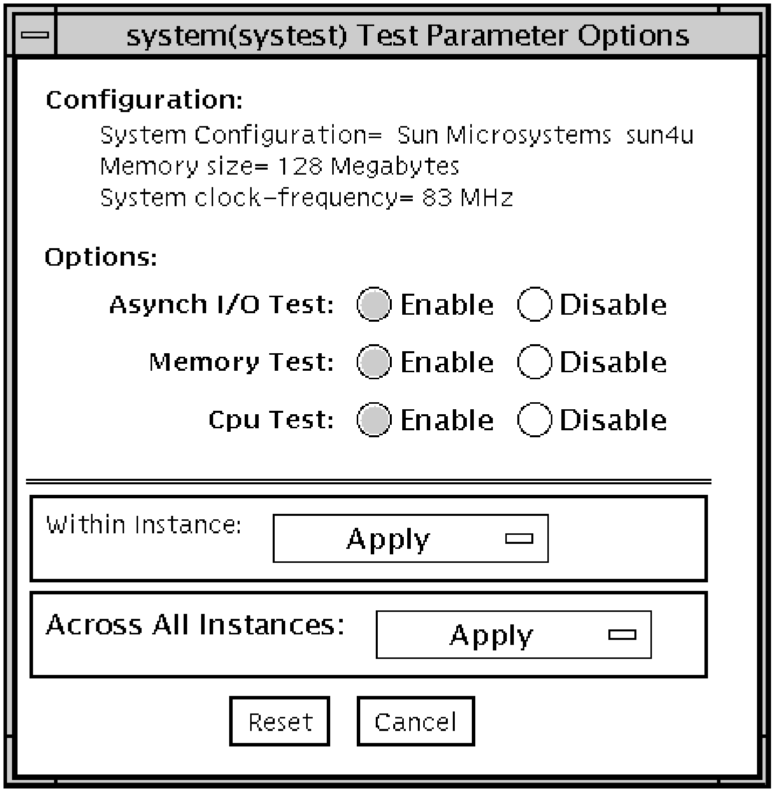


FIGURE 59-1 `systest` Test Parameter Options Dialog Box

systest Test Modes

TABLE 59-1 systest Test Modes

Test Mode	Supported?	Description
Connection	No	Not supported.
Functional (Offline)	Yes	Runs the full set of tests.

systest Command-Line Syntax

/opt/SUNWvts/bin/systest *standard_arguments*

Note – 64-bit tests are located in the sparcv9 subdirectory: /opt/SUNWvts/bin/sparcv9/*testname*. 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 “32-Bit and 64-Bit Tests” on page 3.

CHAPTER **60**

Tape Drive Test (`tapetest`)

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.

`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 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 may 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 Options dialog box for a device is shown in FIGURE 60-1.

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 2.6, Solaris 7, and Solaris 8 operating environments and compatible releases.

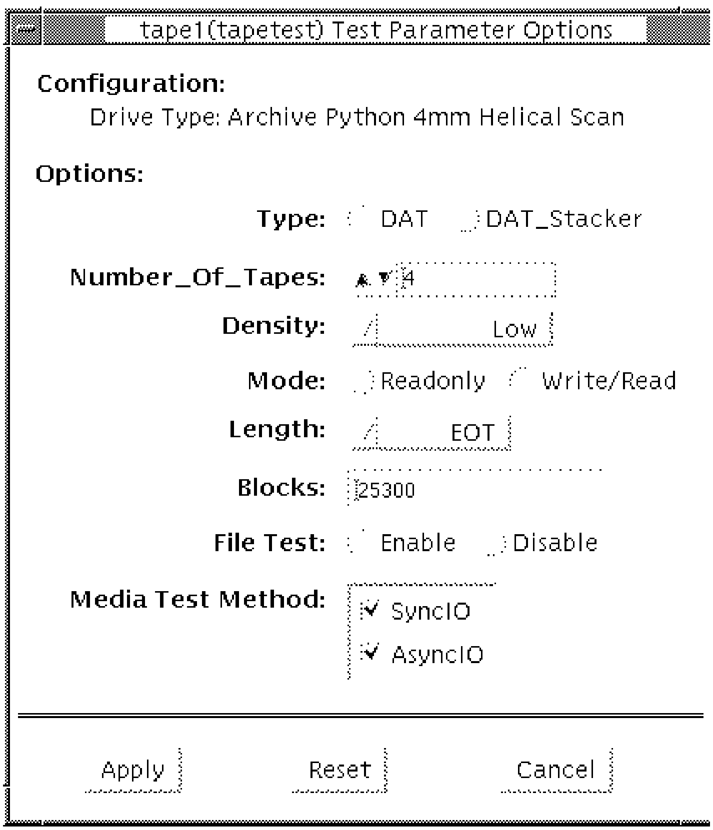


FIGURE 60-1 tapetet Test Parameter Options Dialog Box (4-mm Tape Drive)

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

Note – Selecting non-default options can significantly increase the run time for the test.

Note – The Options dialog box for the 1/4-inch, 1/2-inch, DLT, and 8-mm tape drives differ slightly from FIGURE 60-1.

TABLE 60-1 `tapetest` Options

`tapetest` **Options**	**Description**
Type	Normal tape drive or tape library (stacker).
# 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.

TABLE 60-1 `tapetest` Options

`tapetest` Options	Description
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 `# of blocks` field.
# of Blocks	If you select Specified under the Length option, you must type the number of blocks you want to test.
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. Note1: This option is only available in command line interface mode. Note2: 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 all three modes. It performs different test schemes on the tape device, according to the mode you select.

TABLE 60-2 `tapetest` Test Modes

Test Mode	Supported?	Description
Connection	Yes	`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 (Offline)	Yes	`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 60-3 `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).

TABLE 60-3 `tapetest` Command-Line Syntax *(Continued)*

Argument	Explanation *(Continued)*
`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.
`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*. 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 “32-Bit and 64-Bit Tests” on page 3.

CHAPTER **61**

S24 Frame Buffer Test (`tcxtest`)

Through a series of protocol, memory, acceleration, and colormap tests, `tcxtest` checks the functionality of the S24 Frame Buffer SBus card used on the SPARCstation 5 and checks the FSV (fast SBus video) ASIC on the SPARCstation 4 motherboard.

Note – Disable all screen savers before testing any graphics device. Type **`xset s off`** at a UNIX prompt to disable the Solaris screen saver.

For full instructions on testing frame buffers, see “Testing Frame Buffers” on page 9.

`tcxtest` Test Groups

`tcxtest` has four distinct test groups.

AFX Protocol tests (in 8/16/32/64-bit mode):

- `WRC`

Frame Buffer Memory tests (in 8/16/32/64-bit14 mode):

- `address`
- `constant`
- `random`

Acceleration tests (both User and Raw modes):

- `blit`
- `stip`

Colormap and Cursor tests:

- `cursor` (does not apply to SPARCstation4)
- `colormap`

tcxtest Subtests

TABLE 61-1 tcxtest Subtests

tcxtest Subtests	Description
WRC	By performing multiple writes and reads, and then verifying the results, the WRC test exercises the FIFO inside the S24 chip. The WRC test is composed of these three subtests: `test_afx_alt_wr`, `test_memafx`, and `test_afx_random`. If these tests fail, they print an error message showing the expected and observed data.
Test_afx_alt_wr	This test performs 16 writes to alternative pages, for example, WR (Page1), WR (Page2), WR (Page1+off), WR (Page2+off), and so on. It then reads back the data and compares it with the expected results. This test also writes to the frame buffer space 16 times, followed by a write to a different page in the frame buffer space. The test then reads this data back and verifies it with the expected results.
Test_memafx	The CPU in the SWIFT chip has closely coupled interfaces for the DRAM and the AFX bus. This test checks the arbitration between the two accesses. This test performs a number of alternating writes to the AFX and the CPU memory. After writing to different locations, the test reads and verifies the data. By performing an access across the page boundaries, the test covers both the cached and non-cached accesses.
Test_afx_random	After writing to one page in the DRAM memory, the test performs a few random writes/reads to random locations in the AFX space. The test then writes to a different page in the DRAM space, where it performs random accesses. This test does not perform any data verification, it just checks to see if any of these random accesses caused a time out.
Constant	This test writes a data pattern to the whole memory. This pattern is read back and compared with the expected data. Once the memory fill operation is completed, the test reads the memory back and verifies that the value read is correct.
Address	This test writes a data pattern (which is same as the value of the address) to the whole memory. This pattern is then read back to verify that it is the correct value.

TABLE 61-1 `tcxtest` Subtests

tcxtest Subtests	Description
Random	This test writes a random data pattern to the whole memory. This pattern is read back and compared with the expected data. After the memory fill operation is completed, the test reads the memory and verifies the values read are correct.
Blit	This test has two parts; the raw `blit` test and the user `blit` test. The raw `blit` test draws a 64x64x24 pixel image at the top-left corner of screen. Next it `blits` the image to the screen. The destination images are read back and compared with the original image to verify the raw `blit` operation has run correctly. The user `blit` test draws a 64x64x24 pixel image at the top left corner of screen. It then `blits` the image to the screen. The destination images are read back and compared with the original image. The user `blit` test is the same as the raw `blit` test, except the user `blit` test uses the user data space for the `blit` command.
Stip	This test performs numerous corner cases for stipple. The test writes to the destination with different data values using a stipple operation. The destination data is read back and verified. For the fast SBus video (SV), the following is checked: • Walks 1 through pixel mask. • Walks 1 through ROP bits. • Walks 1 through destination byte. • Walks 1 through IDX byte.
Cursor (does not apply to SPARCstation 4)	This test performs a data register regression test. It writes a walking 1 pattern to the cursor data registers. The data is then read back and verified with the expected results. The test is repeated using a walking 0 as the data pattern
Colormap	Loads all locations in the colormap with varying values of RGB. Note: If the system being tested has a monochrome or greyscale monitor, visual color problems are undetectable.

`tcxtest` Options

To reach the 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 may not include the device appropriate to this test. Refer to the *SunVTS User's Guide* for more details.

FB locking is the only test parameter option for this test. See "Testing Frame Buffers" on page 9 for details.

1. **Click enable or disable to configure frame buffer locking. etc.**

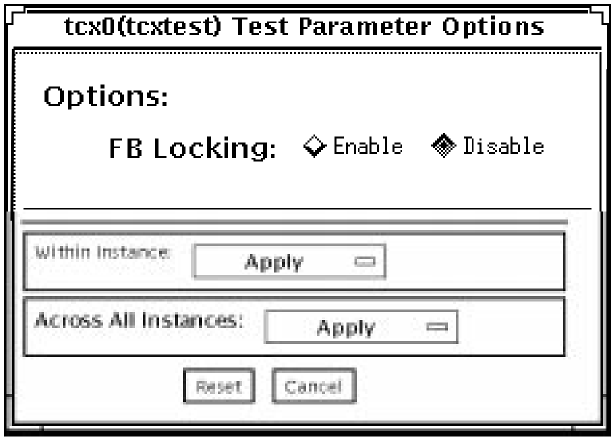


FIGURE 61-1 `tcxtest` Test Parameter Options Dialog Box

`tcxtest` Test Modes

TABLE 61-2 `tcxtest` Test Modes

Test Mode	Supported?	Description
Connection	No	Not supported.
Functional (Offline)	Yes	Runs the appropriate subtests for the hardware platform being tested.

`tcxtest` Command-Line Syntax

`/opt/SUNWvts/bin/tcxtest` *standard_arguments* **`-o`**
`dev=`*device_name***`,lock=`***E(nable)* | *D(isable)***`, X=`***bit_mode***`,T=`***test***`,S=`***[dfb8, dfb24, dfb32]*

TABLE 61-3 `tcxtest` Command-Line Syntax

Argument	Explanation
`dev=`*device_name*	Specifies the filename of the device to be tested, for example, `dev=tcx0`.
`lock=` *E(nable)* | *D(isable)*	Enables or disables the window system locking option. See "Testing Frame Buffers" on page 9 for details. Do not use when device is the window system display.

TABLE 61-3 `tcxtest` Command-Line Syntax *(Continued)*

Argument	Explanation
X=*bit_mode*	Specifies the data transfer size. Supported values are: • `8`byte • `16`short • `32`long • `64`double word
T=*test*	Specifies a particular test. To specify an individual test, replace *test* with: • `a`=Address • `c`=Constant • `r`=Random • `b`=Blit • `s`=Stipple • `h`=Cursor • `w`=WRC Note: When you select either the `Blit` or `Stipple` test, both the User and Raw mode tests are run.
S=*[dfb8, dfb24, dfb32]*	Specifies which frame buffer memory space to use. • `-dfb8`—Dumb frame buffer 8-bit space. Memory is accessed only by bytes. • `-dfb24`—Dumb frame buffer 24-bit space. Memory is accessed only by 24-bit reads and writes. • `-dfb32`—Dumb frame buffer 8-bit space. Memory is accessed by 8-bit reads and writes.

Note – 64-bit tests are located in the `sparcv9` subdirectory: `/opt/SUNWvts/bin/sparcv9/`*testname*. 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 “32-Bit and 64-Bit Tests” on page 3.

CHAPTER **62**

Sun USB Keyboard Test (`usbkbtest`)

`usbkbtest` verifies whether the keyboard(s) attached to the USB bus are USB compliant. The test will flash the LEDs of a compliant keyboard.

`usbkbtest` Options

To reach the 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 may not include the device appropriate to this test. Refer to the *SunVTS User's Guide* for more details.

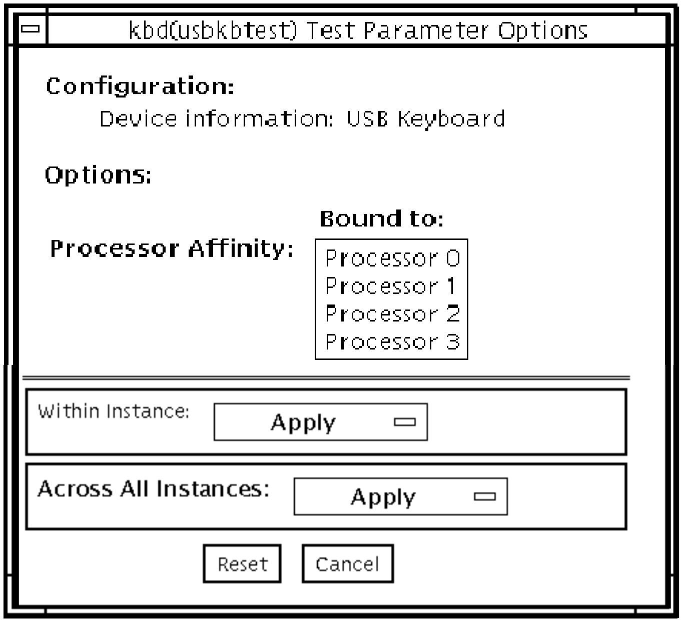


FIGURE 62-1 `usbkbtest` Test Parameter Options Dialog Box

`usbkbtest` Test Modes

TABLE 62-1 `usbkbtest` Test Modes

Test Mode	Supported?	Description
Connection	Yes	Runs the full test.
Functional (Offline)	Yes	Runs the full test.

`usbkbtest` Command-Line Syntax

`/opt/SUNWvts/bin/usbkbtest` *standard_arguments*
`-o dev=`*kbd* | *usb/hid***`n`**

TABLE 62-2 `usbkbtest` Command-Line Syntax

Argument	Description
dev=*kbd* | *usb/hid***n**	Specifies what kind of keyboard is being tested. Use *kbd* for console keyboards and *usb/hid***n** for other keyboards, where **n** is the instance number.

CHAPTER 63

Virtual Memory Test (`vmemtest`)

The `vmemtest` checks virtual memory; that is, it tests the combination of physical memory and the swap partitions of the disk(s).

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

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 the test 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. You should use the `vmemtest` swap space `reserve` option when non-SunVTS test processes are started after SunVTS testing has started. See “Swap Space Requirements” in the *SunVTS User’s Guide* for a complete discussion of swap space requirements.

vmemtest Options

To reach the 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 may not include the device appropriate to this test. Refer to the *SunVTS User's Guide* for more details.

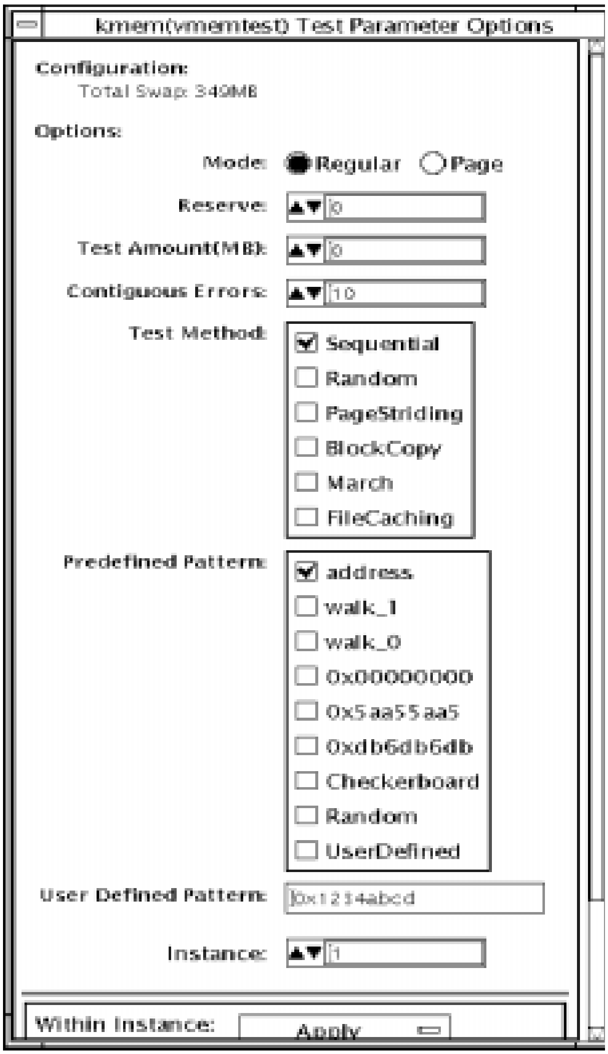


FIGURE 63-1 `vmemtest` Test Parameter Options Dialog Box

TABLE 63-1 `vmemtest` Options

vmemtest Options	Description
Mode	Two modes are available: • Regular mode tests the amount of memory and is limited by the amount of physical memory available to the system under test. • 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. When the Stress Test Execution option is selected, the `vmemtest` allocates the entire assigned system memory (through `valloc`), writing from beginning to end. The memory is then read back and compared with the original pattern, one long word at a time.
Reserve	The Reserve option specifies the amount of memory to reserve from being tested by `vmemtest`. The reserved space is used for other processes running concurrently with the or SunVTS tests. The Reserve option can be used to reserve memory in addition to the default. This option applies only to a specific instance. Trying to reserve more memory than what is assigned to be tested 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. Specifying a number greater than the available memory, can cause `vmemtest` or other running tests to fail.
`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. It indicates the amount of virtual memory found, rounded up to the nearest Kbyte.
Contiguous Errors	Specifies the number of memory errors that occur on successive memory locations before testing stops.

TABLE 63-1 `vmemtest` Options

vmemtest Options	Description
Test Method	• The default selection is the Sequential test. The whole memory is tested from the beginning address to the end address in a sequence. • Address Random test: Randomly selects memory addresses to test. • Page Striding test: Non-contiguous memory test, implemented sequentially and non-sequentially. —Sequential test: tests from the first page to the last page, withn a specified test range. Only one word is tested per page. —Non-sequential test: tests randomly from first to last page, within a specified memory range. Goes back and forth 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 0x00000001 • `walk_0`—uses a pattern that starts with 0x7fff7fff through 0x7fff7ffe • `0x00000000`—uses all ones and zeros for testing • `0x5aa55aa5`—uses 0x5aa55aa5 and 0xa55aa55a patterns • `0xdb6db6db`—uses 0xdb6db6db and 0x24924924 patterns • `Checkerboard`—uses 0x55555555 and 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`.
Instance	Specifies how many copies of the `vmemtest` test to run.

vmemtest Test Modes

TABLE 63-2 vmemtest Test Modes

Test Mode	Supported?	Description
Connection	No	Not supported.
Functional (Offline)	Yes	Runs the full set of tests.

In Offline Functional 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. When there is a miscompare on recomparison, an attempt is made to convert the virtual address to the physical address if the SunVTS diagnostic driver is installed.

vmemtest Command-Line Syntax

/opt/SUNWvts/bin/vmemtest *standard_arguments* **-o mode=***type***,**
reserve=*n***,amount=***n***,bdinfo=***n***,cerr=***number***,type=***n***,pp=***pattern***,up=***hex_address*

TABLE 63-3 vmemtest Command-Line Syntax

Argument	Explanation
mode=Page|Regular	Specifies which mode of the vmemtest to run. Choose: • 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 Mbytes of virtual memory to reserve in addition to the default amount.
amount=*n*	Specifies the number of Mbytes of memory to be tested instead of the default (maximum) amount.
bdinfo=*n*	Provides the board number information for all the CPU/memory boards in the system. For example, if board 0 and board 5 have memory, then the bdinfo=33 (2**5+2**0).

TABLE 63-3 `vmemtest` Command-Line Syntax

Argument	*(Continued)* Explanation
`cerr=`*n*	Specifies the maximum number of contiguous errors to be dumped when a memory error occurs.
`type=`*seq+rand+page+ block+file*	• seq—Uses Sequential test method • rand—Uses Random Address test method • page—Uses Page Striding test method • block—Uses Block Copy method • file—Uses File Caching test method
`pp=`*pattern*	Specifies the test pattern to use for testing. Valid pattern entries are: • `Address`—uses the virtual addresses of the tested memory locations. • `walk_1`—uses a pattern that starts with 0x80000000 through 0x00000001. • `walk_0`—uses a pattern that starts with 0x7fff7fff through 0x7fff7ffe. • `0x00000000`—uses all ones and zeros for testing. • `0x5aa55aa5`—uses `0x5aa55aa5` pattern. • `0xdb6db6db`—uses `0xdb6db6db` pattern. • `Checkerboard`—uses `0x55555555` and `0xaaaaaaaa` patterns. • Random—uses a random data pattern. • `UserDefined`—uses the pattern that is specified in the User Defined Pattern area (see below).
`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*. 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 “32-Bit and 64-Bit Tests” on page 3.

APPENDIX **A**

Loopback Connectors

Loopback connectors are designed for the testing of communication ports. They 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.

Table A-1 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

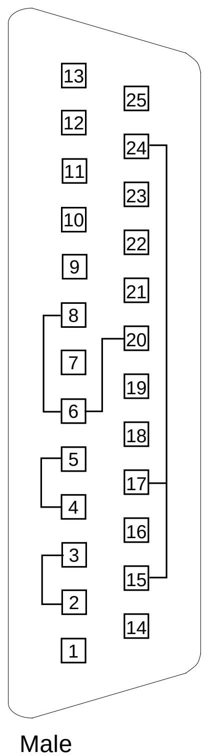
TABLE A-1 Loopback Connector Pin Assignments *(Continued)*

Signal Description	EIA	CCITT #	RS-449 "A"	RS-449 "B"	DIN 8 8-pin round	DB9 9-pin	DB25 25-pin	Direction	Alpha ID
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)	SC F	122	NC*	NC*	NC*	NC*	12	Input	SCF
Secondary Clear to Send (CTSb)	SC B	121	NC*	NC*	NC*	NC*	13	Input	SCB
Secondary Transmit Data (TxDb)	SB A	118	NC*	NC*	NC*	NC*	14	Output	SBA
Secondary Receive Data (RxDb)	SB B	119	NC*	NC*	NC*	NC*	16	Input	SBB
Secondary Request to Send (RTSb)	SC A	120	NC*	NC*	NC*	NC*	19	Output	SCA

*NC = No connection

25-Pin RS-232 Loopback Plug

The RS-232 and RS-423 single-port loopback plug is a specially wired male DB-25 connector. It is plugged in to a serial port in the back of the system under test.



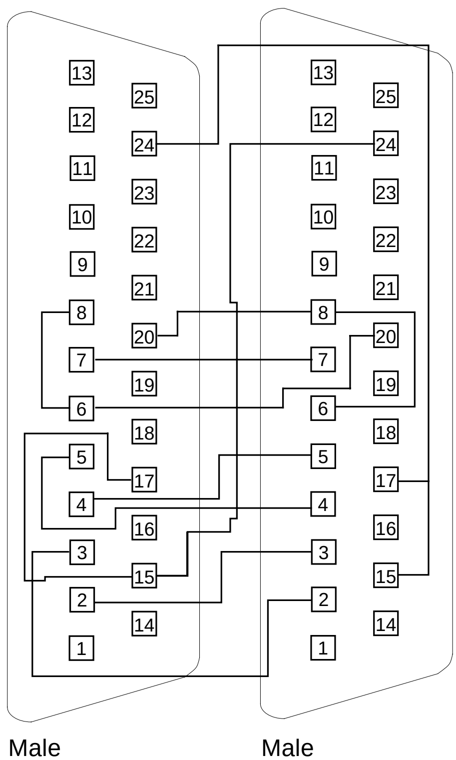
Connect:

First connector	to	Second connector
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

Use these wiring instructions for 25-pin RS-232 and RS-423 port to 25-pin RS 232 and RS 423 port loopback cables (two DB-25 connections). It is plugged into a pair of serial ports in the back of the system under test. Both connectors are male.



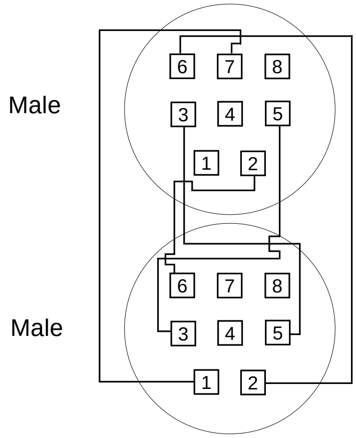
Connect:

First connector	to	Second connector
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

Use these wiring directions for 8-pin round DIN RS-232 port to RS-423 to 8-pin round-DIN RS-232 and RS-423 port loopback cable. Both connectors are male.



Connect:

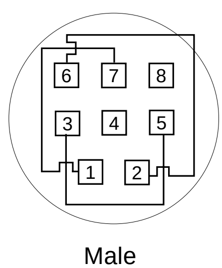
First connector	to	Second connector
Pin 3		Pin 5
Pin 5		Pin 3
Pin 6		Pin 2
Pin 2		Pin 6
Pin 7		Pin 1

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

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

8-Pin Loopback Plug

Use these wiring directions for male 8-pin round-DIN RS-232 and RS-423 single-port loopback plugs.



Connect:

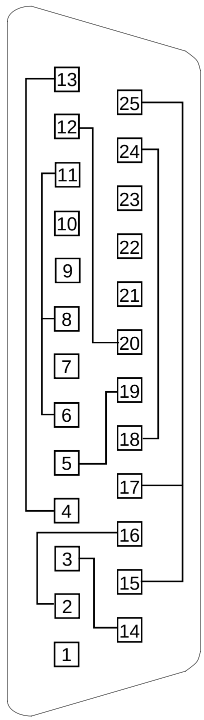
First connector	to	Second connector
Pin 3		Pin 5
Pin 6		Pin 2
Pin 1		Pin 7

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

Use these wiring directions for a 25-pin Port A to Port B loopback plug for most systems.



Male

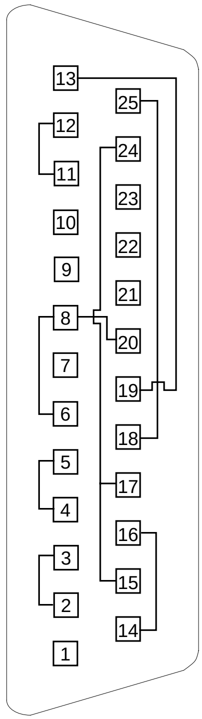
Connect:

First connector	to	Second connector
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 communication port to connect it to peripherals, use these wiring instructions for making a male 25-pin loopback plug for that communication port.



Male

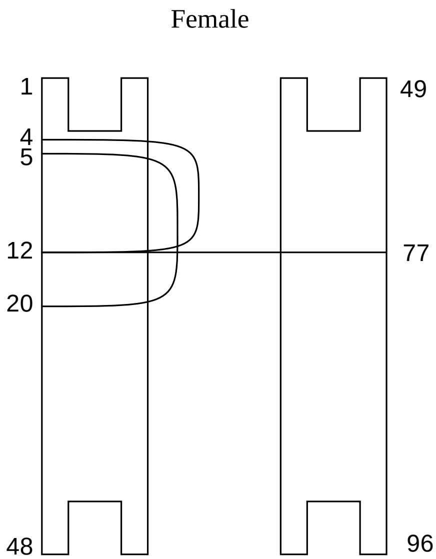
Connect:

First connector	to	Second connector
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

This 96-pin connector can be ordered from Sun (part number 370-1366).



Materials:
PCR-E96FA(1)
PCS-E96LKPA(1)
3751 Metal Plug(1)
(9563K42)
AWG28 Madison Cable(8” long)
UL/CSA Approved

Connect:

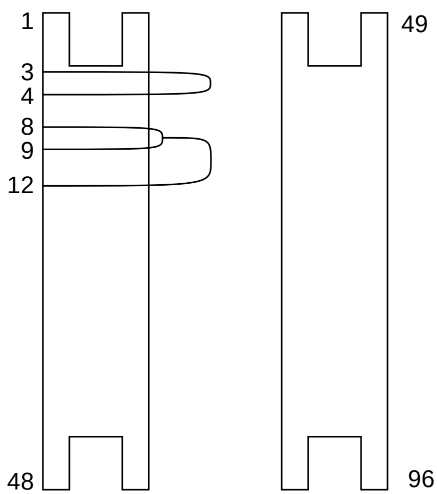
First connector	to	Second connector
Pins 4 and 12		Pin 77
Pin 5		Pin 20
Pins 6		Pin 36
Pin 10		Pin 11
Pins 13		Pin 16
Pin 14		Pin 15
Pin 18		Pin 19
Pin 21		Pin 24
Pin 28		Pin 60
Pin 29		Pin 68
Pin 30		Pin 34
Pin 37		Pin 40
Pin 38		Pin 39
Pin 42		Pin 43
Pin 45		Pin 48
Pin 46		Pin 47
Pin 52		Pin 78
Pin 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

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).

Female Special



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.

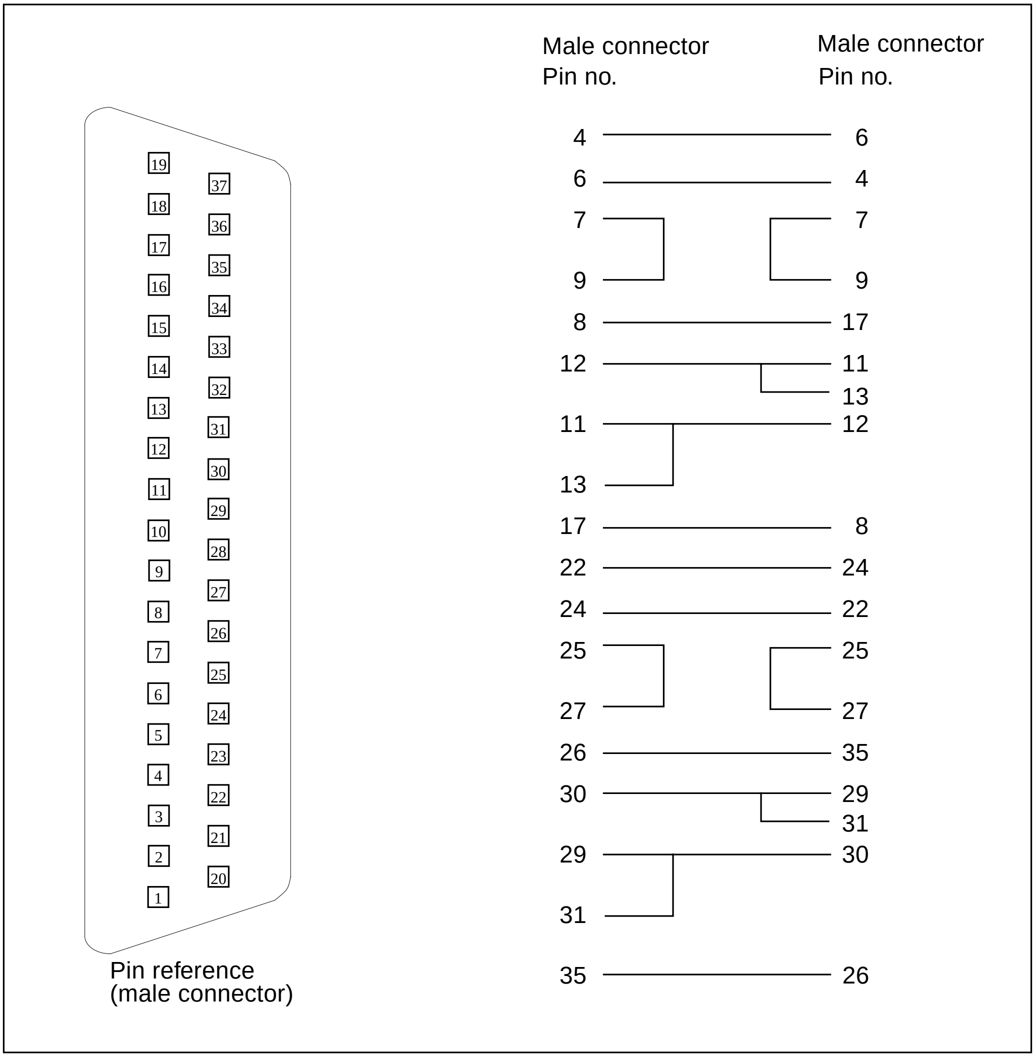
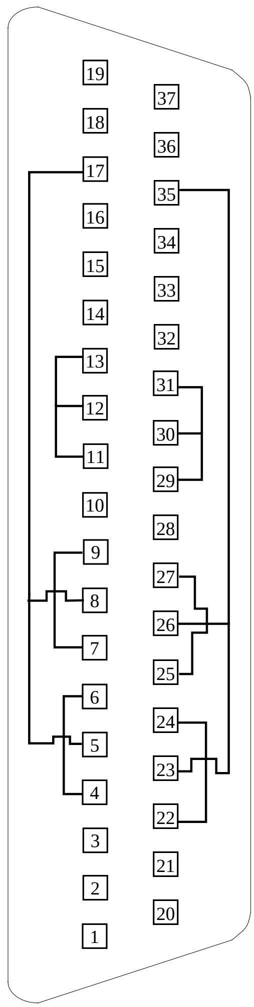


FIGURE A-9 37-Pin RS-449 Loopback Cable Wiring Diagram

37-Pin RS-449 Loopback Plug

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



Male

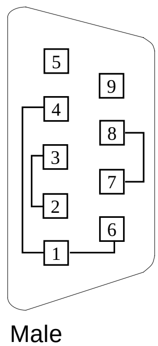
Connect pin:

4 to 6
5 to 8 to 17
7 to 9
11 to 12 to 13
22 to 24
23 to 26 to 35
25 to 27
29 to 30 to 31

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.



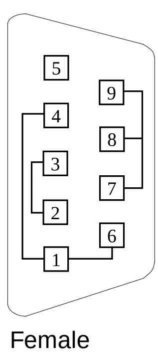
Connect:

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

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`.



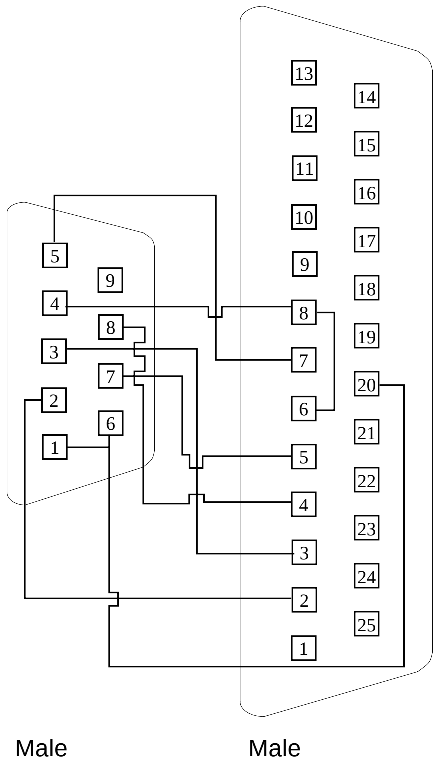
Connect:

First connector	to	Second connector
Pin 2		Pin 3
Pin 1		Pins 4 and 6
Pin 7		Pins 8 and 9

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.



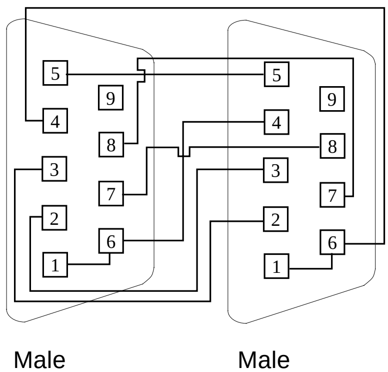
Connect:

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 connectors 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

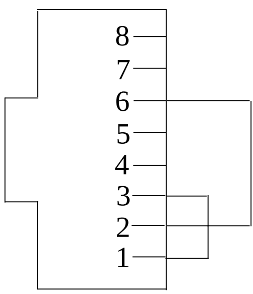
Please note that this cable has no Sun part number assigned to it.

NT to TE Loopback Cable

Using two standard RJ45 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

Use these wiring instructions for standard RJ-45 connectors. This loopback cable is used in `netlbtest` for eri devices.



Connect:

First connector	to	Second connector
Pin 1		Pin 3
Pin 2		Pin 6

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

Index

D

E

K

L

M

N

T

U

V

W

Z