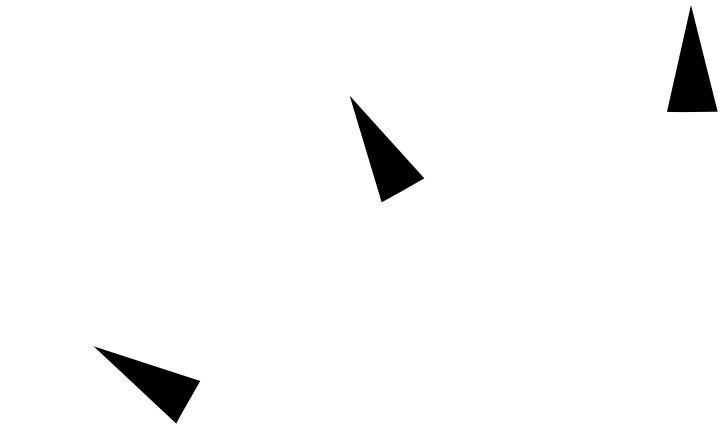
SunVTS™ 2.0 Test Reference Manual



Sun Microsystems Computer Company
2550 Garcia Avenue
Mountain View, CA 94043 USA
415 960-1300 fax 415 969-9131

Part No: 802-5330-10
Revision A, May 1996

Please
Recycle

Contents

Preface xxv

1. Introduction 1

Accessing SunVTS 1

Hardware and Software Requirements 6

OPEN LOOK Software Requirements 6

Testing Multiple Frame Buffers 7

Remote Testing 7

2. `audio` Test 9

`audio` Subtests 9

`audio` Options 10

`audio` Test Modes 17

`audio` Command Line Syntax 18

`audio` Error Messages 20

3. `bpptest` 25

Bidirectional Parallel Port Printer Test (`bpptest`) 25

Printer Test Hardware and Software Requirements 26
bpptest Options. 26
bpptest Test Modes. 27
bpptest Command Line Syntax. 28
bpptest Error Messages 28
4. cdtest 31
Compact Disc Test (cdtest) 31
cdtest Options. 32
cdtest Test Modes 33
cdtest Command Line Syntax. 34
cdtest Error Messages 35
5. cg14test 39
Color Graphics Frame Buffer Test (cg14test) 39
cg14test Groups 40
MDI and VBC Chip Control Registers (test group 1) 41
Memory Chips (test group 2) 42
MDI Chip Cursor Registers (test group 3) 44
MDI Chip CLUT Registers (test group 4) 45
DAC Chip Registers (test group 5) 45
MDI Chip XLUT Registers (test group 6) 45
CG14 Display (visual only) (test group 7) 46
MDI Chip Test Mode Readback [TMRB] (test group 8) . . . 46
Driver IOCTLs (test group 9) 46
cg14test Options 47

cg14test Command Line Syntax 48
cg14test Test Modes..... 49
cg14test Error Messages 49
6. cg6..... 55
Frame Buffer, GX, GX+ and TGX Options Test (cg6)..... 55
cg6 Options 57
cg6 Test Modes 57
cg6 Command Line Syntax 58
cg6 Error Messages..... 58
7. disktest..... 69
disktest Test Options..... 71
disktest Test Modes..... 74
disktest Command Line Syntax 75
disktest Error Messages 76
8. ecpptest..... 93
ECP 1284 Parallel Port Printer Test (ecpptest) 93
Printer Test Hardware and Software Requirements 93
ecpptest Subtests 94
ecpptest Options 95
ecpptest Test Modes..... 95
ecpptest Command Line Syntax 96
ecpptest Error Messages 96
9. fbtest 103
Frame Buffer Test (fbtest) 103

`fbtest` Command Line Syntax 104
`fbtest` Test Modes 104
`fbtest` Error Messages 105
10. `ffbtest` 109
Fast Frame Buffer Test (`ffbtest`) 109
`ffbtest` Options 110
3DRAM Test Description 111
3DRAM Logic Test Description 111
RAMDAC Test Description 112
Rendering Pipeline Test Description 113
Fast Fill / Vertical Scroll Test Description 113
Pixel Process Test Description 113
Picking Test Description 114
Arbitration Test Description 114
Stereo Test Description 114
`ffbtest` Test Modes 115
`ffbtest` Command Line Syntax 117
`ffbtest` Error Messages 118
11. `fputest` 125
Floating Point Unit Test (`fputest`) 125
`fputest` Options 126
`fputest` Test Modes 126
`fputest` Command Line Syntax 127
`fputest` Error Messages 127

12. isdntest 205
Dual Basic Rate ISDN (DBRI) Chip (isdntest) 205
isdntest Options 209
isdntest Test Modes 210
isdntest Command Line Syntax 210
isdntest Error Messages 211
13. leotest 213
ZX and TZX Graphics Accelerator Test (leotest) 213
leotest Options 214
leotest Subtests 216
Direct Port Tests 216
Accelerator Port Tests 217
leotest Test Modes 220
leotest Command Line Syntax 221
leotest Command Line Examples 222
leotest Error Messages 222
14. lpvitest 231
Printer Test Hardware and Software Requirements 231
lpvitest Options 232
lpvitest Test Mode 233
lpvitest Command Line Syntax 234
lpvitest Error Messages 234
15. mptest 241
Multiprocessor Test (mptest) 241

mptest Options 242
mptest Test Modes 244
mptest Command Line Syntax 244
mptest Error Messages 245
16. nettest 251
Network Hardware Test (nettest) 251
nettest Options 253
nettest Test Modes 254
nettest Command Line Syntax 255
nettest Error Messages 255
17. pcsertest 259
PCMCIA Modem Card Test (pcsertest) 259
pcsertest Options 260
pcsertest Test Mode 260
pcsertest Command Line Syntax 260
pcsertest Error Messages 261
18. plntest 263
SPARCstorage Array Controller Test (plntest) 263
plntest Options 264
plntest Test Modes 265
plntest Command Line Syntax 266
Probing for SSA Controller Devices 266
plntest Error Messages 267
19. pmem 269

Physical Memory Test (pmem) 269
pmem Options 269
pmem Test Modes 270
pmem Command Line Syntax 271
pmem Error Messages 271
20. pstest 275
Prestoserve Test (pstest) 275
pstest Options 276
pstest Test Modes 277
pstest Command Line Syntax 277
pstest Error Messages 277
21. rtvctest 281
SunVideo Test (rtvctest) 281
rtvctest Subtests 281
rtvctest Options 284
rtvctest Command Line Syntax 286
rtvctest Error Messages 286
22. spdtest 291
NeWSprinter Test (spdtest) 291
spdtest Options 292
spdtest Test Mode 293
spdtest Command Line Syntax 294
spdtest Error Messages 295
23. spif 301

Serial Parallel Controller Test (spif) 301
spif Hardware Requirements. 301
spif Options 302
spif Test Modes 305
spif Command Line Syntax 305
spif Error Messages. 306
24. sptest 309
Serial Ports Test (sptest). 309
Syncloop Testing 309
Syncloop Testing Software Requirements. 310
sptest Options. 311
sptest Test Modes 313
sptest Command Line Syntax. 314
sptest Error Messages 315
25. sunbuttons. 319
SunButtons Test (sunbuttons). 319
sunbuttons Command Line Syntax 321
sunbuttons Test Modes 321
sunbuttons Error Messages. 321
26. sundials 323
SunDials Test (sundials) 323
sundials Test Modes. 325
sundials Command Line Syntax 325
sundials Error Messages 325

27. `sunlink` **327**
HSI/S Boards Test (`sunlink`) 327
`sunlink` Configurations..... 328
`sunlink` Command Line Syntax..... 329
`sunlink` Test Modes..... 330
`sunlink` Loopback Connectors..... 330
`sunlink` Error Messages 330
28. `sxtest` **333**
Pixel Processor Test (`sxtest`) 333
`sxtest` Module Descriptions 336
Display (Module0)..... 336
MUL (Module1)..... 337
ALU (Module2) 338
ROP (Module3) 338
LOGIC (Module4) 339
SHIFT (Module5)..... 340
COMP (Module6)..... 340
MISC (Module7)..... 341
MADR (Module8) 342
SMCALL (Module9) 343
MCNT (Module10) 344
GRIF (Module11) 345
REGF (Module12) 346
`sxtest` Test Modes..... 347

sxtest Command Line Syntax 348
sxtest Error Messages 348
29. systest 351
systest Options 351
systest Test Modes 352
systest Command Line Syntax 352
systest Error Messages 352
30. tapetest 355
Tape Drive Test (tapetest) 355
tapetest Options 355
tapetest Test Modes 361
tapetest Command Line Syntax 362
tapetest Error Messages 363
31. tcxtest 369
S24 Frame Buffer Test (tcxtest) 369
tcxtest Subtests 370
tcxtest Options 373
FB Locking 373
tcxtest Test Modes 373
tcxtest Command Line Syntax 374
tcxtest Error Messages 375
32. vmem 391
Virtual Memory Test (vmem) 391
Swap Space Consideration 391

vmem Options 392
vmem Test Modes 393
vmem Command Line Syntax 394
vmem Error Messages 394
33. xbtest 397
SBus Expansion Subsystem (xbtest) 397
xbtest Options 399
xbtest Test Modes 400
xbtest Command Line Syntax 400
xbtest Error Messages 401
A. Loopback Connectors 405
25-Pin RS-232 Loopback Plug 407
25-pin RS-232 Port-to-Port Loopback Cable 407
8-Pin to 8-Pin Loopback Cable 408
8-Pin Loopback Plug 409
25-pin Port A-to-Port B Loopback Plug 410
25-pin Port A-to-A Port B-to-B Loopback Plug 410
96-Pin Female Loopback Connector 411
96-Pin Female Special Loopback Connector 413
37-Pin RS-449 Loopback Cable 414
37-Pin RS-449 Loopback Plug 415
9-pin Male Single-port Loopback Plug 416
9-pin Female Single-port Loopback Plug 416
9-Pin to 25-Pin Port-to-Port Loopback Cable 417

9-Pin to 9-Pin Port-to-Port Loopback Cable 417
NT to TE Loopback Cable 418
Index 419

Figures

Figure 2-1 `audio` Test Option Menu for `audioamd(7)` 11

Figure 2-2 `audio` Test Option Menu for speakerbox `dbri(7)` `audio` .. 12

Figure 2-3 `audio` Test Option Menu for SPARCstation LX `dbri(7)` with no speakerbox attached 13

Figure 2-4 `audio` Test Option Menu for on-board `dbri(7)` for SPARCstation 20 and S240 14

Figure 2-5 `audio` Test Option Menu for `audiocs(7)` 15

Figure 2-6 `audio` Test Option Menu for `audiocs`(7) with Internal Loopbacks 16

Figure 3-1 `bpptest` Option Menu 26

Figure 4-1 `cdtest` Option Menu 32

Figure 5-1 `cg14test` Option Menu 47

Figure 6-1 `cg6` Options Menu 57

Figure 7-1 `disktest` Configuration and Options Menu 71

Figure 8-1 `ecpptest` Option Menu 95

Figure 9-1 `fbtest` Option Menu 103

Figure 10-1 `ffbtest` Option Menu 110

Figure 11-1 `fputest` Option Menu 126
Figure 12-1 `isdntest` Local Loopback Subtest 206
Figure 12-2 `isdntest` Remote Loopback Subtest 207
Figure 12-3 `isdntest` Read / Write Subtest. 208
Figure 12-4 `isdntest` Data Path Subtest. 209
Figure 12-5 `isdntest` Options Menu 209
Figure 13-1 `leotest` Option Menu 215
Figure 14-1 `lpvitest` Option Menu 232
Figure 15-1 `mptest` Option Menu. 242
Figure 16-1 `nettest` Option Menu 253
Figure 17-1 `pcsertest` Option Menu 260
Figure 18-1 `plntest` Option Menu 264
Figure 19-1 `pmem` Option Menu 269
Figure 19-2 UltraSPARC Server Option Menu 270
Figure 20-1 `pstest` Option Menu. 276
Figure 21-1 `rtvctest Option Window` 284
Figure 22-1 `spdtest` (NeWSprinter) Option Menu 292
Figure 23-1 `spif` Option Menu 302
Figure 24-1 `sptest` Option Menu. 311
Figure 25-1 `sunbuttons` Test Menu 320
Figure 26-1 `sundials` Test Menu 324
Figure 27-1 `sunlink` Option menu. 328
Figure 28-1 `sxtest` Option Menu (CMEM:0MB, none, `cg14` board) 334
Figure 28-2 `sxtest` Option Menu (CMEM:4MB, cg14:4MB) 335
Figure 29-1 `systest` Option Menu 351

Figure 30-1 `tapetest` Option Menu (8mm tape drives). 356
Figure 30-2 `tapetest` Option Menu (half-inch front-load tape drives). . 357
Figure 30-3 `tapetest` Option Menu (quarter-inch tape drives) 358
Figure 30-4 `tapetest` Option Menu (4mm tape drives). 359
Figure 31-1 `tcxtest` Option Menu 373
Figure 32-1 `vmem` Option Menu 392
Figure 33-1 `xbtest` Option Menu. 399
Figure A-1 25-pin RS-232 Loopback Plug 407
Figure A-2 25-pin RS-232 Port-to-Port Loopback Cable 408
Figure A-3 8-Pin to 8-Pin Loopback Cable. 409
Figure A-4 8-Pin Loopback Plug 409
Figure A-5 Port A-to-Port B Loopback Plug 410
Figure A-6 Port A-to-A, Port B-to-B Loopback Plug 411
Figure A-7 96-Pin Female Loopback Connector 412
Figure A-8 96-Pin Female Special Loopback Connector 413
Figure A-9 37-Pin RS-449 Loopback Cable 414
Figure A-10 37-Pin RS-449 Loopback Plug 415
Figure A-11 9-Pin Male Single-port Loopback Plug. 416
Figure A-12 9-Pin Female Single-port Loopback Plug. 416
Figure A-13 9-Pin to 25-Pin Port-to-Port Loopback Cable 417
Figure A-14 9-Pin to 9-Pin Port-to-Port Loopback Cable 418

Tables

Table P-1 Typographic Conventions xxvi

Table P-2 Shell Prompts xxvii

Table P-3 SunVTS Documents. xxvii

Table 1-1 Standard SunVTS Command Line Arguments. 2

Table 1-2 SunVTS Test-Specific Arguments 3

Table 1-3 Test Modes Available for each SunVTS Test. 4

Table 5-1 `cg14test` NTA Testing Patterns 42

Table 21-1 `rtvctest` Verification Modules. 282

Table 24-1 Loopback Settings 312

Table A-1 Pin Connections for Loopback Plugs 405

Preface

The Sun Validation and Test Suite (SunVTS) product is a system exerciser that verifies the configuration, functionality, and reliability of hardware controllers and devices. SunVTS is primarily used from a graphical user interface (GUI), which may be either OPEN LOOK or Common Desktop Environment™ (CDE). This book contains descriptions of SunVTS tests that run on machines with SPARC® architectures. The descriptions include specific test options, procedures, and error messages.

The primary audience for this book is hardware testing and verification personnel. This book can also be used by developers or sophisticated users who want to run SunVTS diagnostic applications in a test environment.

How This Book Is Organized

This manual is organized as follows:

Chapter 1, "Introduction" describes how the tests are grouped and directs you to the glossary for definitions of unfamiliar terms. You are told how to access SunVTS, the hardware and software requirements for running SunVTS, how to test frame buffers, and how to do remote testing.

Chapters 2 through 33, describe the SunVTS tests, options, command line syntax, other applicable test modes, and error messages.

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

UNIX Commands

This document may not include specific software commands or procedures. Instead, it may name software tasks and refer you to operating system documentation or the handbook that was shipped with your new hardware.

The type of information that you might need to use references for includes:

- Shutting down the system
- Booting the system
- Configuring devices
- Other basic software procedures

See one or more of the following:

- *Solaris 2.x Handbook for SMCC Peripherals* contains Solaris™ 2.x software commands.
- On-line AnswerBook™ for the complete set of documentation supporting the Solaris 2.x software environment.
- Other software documentation that you received with your system.

Typographic Conventions

The following table describes the typographic changes used in this book.

Table P-1 Typographic Conventions

Typeface or Symbol	**Meaning**	**Example**
`AaBbCc123`	The names of commands, files, and directories; on-screen computer output	Edit your `.login` file. Use `ls -a` to list all files. `machine_name% You have mail.`

Table P-1 Typographic Conventions

Typeface or Symbol	Meaning	Example
AaBbCc123	What you type, contrasted with on-screen computer output	`machine_name% su` `Password:`
AaBbCc123	Command-line placeholder: replace with a real name or value	To delete a file, type `rm` *filename*.
AaBbCc123	Book titles, new words or terms, or words to be emphasized	Read Chapter 6 in the *User's Guide*. These are called *class* options. You *must* be root to do this.

Shell Prompts

The following table shows the default system prompt and superuser prompt for the C shell, Bourne shell, and Korn shell.

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

The following table lists other SunVTS documents and related manuals:

Table P-3 SunVTS Documents

Document Title	Part Number
SunVTS 2.0 Quick Reference Card	802-5329
SunVTS 2.0 User's Guide	802-5331

Ordering Sun Documents

The SunDocs Order Desk is a distribution center for Sun Microsystems technical documents. You can use major credit cards and company purchase orders. You can order documents in the following ways:

Country	Telephone	Fax
United States	1-800-873-7869	1-800-944-0661
United Kingdom	0-800-89-88-88	0-800-89-88-87
France	05-90-61-57	05-90-61-58
Belgium	02-720-09-09	02-725-88-5
Luxembourg	32-2-720-09-09	32-2-725-88-5
Germany	01-30-81-61-91	01-30-81-61-92
The Netherlands	06-022-34-45	06-022-34-46
Sweden	020-79-57-26	020-79-57-27
Switzerland	155-19-26	155-19-27
Japan	0120-33-9096	0120-33-9097
World Wide Web: `http://www.sun.com/sunexpress/`		

Sun Welcomes Your Comments

Please use the *Reader Comment Card* that accompanies this document. We are interested in improving our documentation and welcome your comments and suggestions.

If a card is not available, you can email or fax your comments to us. Please include the part number of your document in the subject line of your email or fax message.

- Email: `smcc-docs@sun.com`
- Fax: SMCC Document Feedback
  1-415-786-6443

Introduction 1

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

SunVTS works from either the Common Desktop Environment™ (CDE) user interface or the OPEN LOOK™ (OL) user interface, which lets you set test parameters quickly and easily while running the diagnostic tests. The sample screens and menus in this manual are of SunVTS using the OPEN LOOK user interface.

This manual describes SunVTS Version 2.0, which is on the *SMCC Updates CD*. The default installation directory for SunVTS is `/opt/SUNWvts`. When you are installing SunVTS, you can specify a different directory to install the software.

Accessing SunVTS

You can access SunVTS from various interfaces: CDE, OL, or the TTY interface. SunVTS tests can be run from a shell command line, using the command line syntax for each test. The SunVTS kernel probes for hardware devices installed on your system.

Graphical User Interfaces (GUIs)

SunVTS GUIs let you select tests and test options by pointing and clicking with a mouse button. You can use the CDE or OL interface.

TTY Interface

Using the TTY interface, you can run SunVTS from a terminal or modem attached to a serial port. This feature requires that you use the keyboard instead of using the mouse, and it displays one screen of information at a time. However, it emulates the window system whenever possible.

Command Line Interface

You can also run each of the SunVTS tests individually from a shell command line using the command line syntax. Each test description contains the corresponding command line syntax.

For more information about running individual tests from the command line, refer to the specific test description in this manual.

Standard Command Line Arguments

Different types of command arguments can be applied to a test: generic command arguments (common to all tests), and test-specific command arguments. Because the code for each test defines test-specific arguments, this section only addresses generic command parameters.

The standard usage for all SunVTS tests is:

```
Usage: testname [-scruvdtelnf] [-p number][-i number] [-w number] [-o test specific arguments]
```

The following table defines the standard SunVTS command Line arguments:

Table 1-1 Standard SunVTS Command Line Arguments

Argument	Definition
`-s`	Runs a test in SunVTS mode.
`-c`	Enables a core dump; the test creates a core file if a system crash occurs.
`-r`	Runs on Error; if an error occurs, the test continues the next test sequence instead of exiting.
`-u`	Displays the Usage statement.
`-v`	Runs the test in Verbose mode; the test displays VERBOSE messages that tell more about the testing process.

Table 1-1 Standard SunVTS Command Line Arguments *(Continued)*

Argument	Definition
`-d`	Runs the test in Debug mode; the test displays DEBUG messages to help programmers debug their test code.
`-t`	Runs the test in test function trace mode; the test displays TRACE messages that track down function calls and sequences currently being used by the test code.
`-e`	Runs in stress mode; the test runs under increased system load.
`-l`	Runs in online mode.
`-n`	Runs in connectivity mode.
`-f`	Runs in offline mode.
`-i` *number*	Defines the number of instances for scalable tests.
`-p` *number*	Defines the number of passes.
`-w` *number*	For scalable tests, defines which instance the test is assigned.

Test-Specific Arguments

Test-specific arguments should follow the format specified in the `getsubopt`(3c) man page. There should be at least one test-specific argument, as described in Table 1-2.

Table 1-2 SunVTS Test-Specific Arguments

Argument	Definition
`-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).

Test Modes

SunVTS has several test modes that you can select during testing. The modes are Connectivity, Online, Offline, and Stress. The following table lists the various tests by group, and indicates the test modes available for each test.

Table 1-3 Test Modes Available for each SunVTS Test

Test Name	Connectivity	Online	Offline	Stress
System Board				
audiotest	Yes	Yes	Yes	Yes
fputest	Yes	Yes	Yes	No
mptest	Yes	Yes	Yes	No
pmem	Yes	Yes	Yes	No
systest	No	No	Yes	No
vmem	No	No	Yes	No
Devices				
cdtest	Yes	Yes	Yes	Yes
disktest	Yes	Yes	Yes	No
tapetest	Yes	Yes	Yes	No
Parallel Ports				
bpptest	Yes	Yes	Yes	No
ecpptest	Yes	Yes	Yes	No
lpvitest	No	No	Yes	No
spdtest	No	No	Yes	No
Comm Ports				
spiftest	No	No	Yes	No
sptest	Yes	Yes	Yes	No
sunlink	No	No	Yes	No
Networks				
isdntest	Yes	Yes	Yes	No
nettest	Yes	Yes	Yes	No

Table 1-3 Test Modes Available for each SunVTS Test

Test Name	Connectivity	Online	Offline	Stress
Controllers				
plntest	Yes	Yes	Yes	No
pstest	No	No	Yes	Yes
Frame Buffers				
cg6test	No	No	Yes	Yes
cg14test	No	No	Yes	Yes
fbtest	No	No	Yes	No
ffbtest	No	No	Yes	No
leotest	No	No	Yes	No
sxtest	No	No	Yes	No
tcxtest	No	No	Yes	Yes
Miscellaneous				
pcsertest	No	No	Yes	No
rtvctest	No	No	Yes	No
sundials	No	No	Yes	No
sunbuttons	No	No	Yes	No
xbtest	No	No	Yes	No

Hardware Verification

The SunVTS kernel automatically probes the system kernel for installed hardware devices. Those devices are then displayed on the SunVTS control panel with the appropriate tests and test options. This provides a quick check of your hardware setup.

Hardware and Software Requirements

The SunVTS Version 2.0 software runs on any system with the Solaris 2.5 (or later) operating environment installed. The operating system kernel must be configured to support all peripherals that are to be tested.

OPEN LOOK Software Requirements

You must meet the following requirements to run SunVTS with OPEN LOOK:

- Run Solaris 2.5 operating environment, or later
- Run OPEN LOOK Version 3.3 or later
- Set the correct library path

You may have to set the LD_LIBRARY_PATH variable, depending on the location of the OPEN LOOK directory in your system.

- If you installed or mounted OPEN LOOK files in the default location, `/usr/openwin`, you can ignore this step.
- If OPEN LOOK is installed in a different location, you must specify the location of the OPEN LOOK libraries. Use the following command and substitute the *pathname* variable for the actual path where you installed the OPEN LOOK software:

```
# setenv LD_LIBRARY_PATH pathname
```

♦ **Check the existing LD_LIBRARY_PATH by typing** env**.**

Testing Multiple Frame Buffers

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

- You can test multiple frame buffers on a system at the same time, but only one frame buffer can run the OPEN LOOK software.
- To avoid incorrect test failures, the frame buffer that runs the OPEN LOOK software must have window locking enabled. Any other frame buffers must have window locking disabled.

Caution – If window locking is disabled (unlocked) on frame buffers that are running OPEN LOOK software, the SunVTS tests can return spurious error messages if you move the mouse during testing. Even a slight mouse movement can cause a test to fail.

- By default, SunVTS enables window locking on the console monitor (frame buffers that are pointed by `/dev/fb`).
- If you are running a frame buffer test from a command line, you can disable window locking by specifying a command line argument (see the test command line descriptions in this manual). For example, when running the generic frame buffer test (`fbtest`), use the `lock=e/d` option to disable or enable frame buffer locking. Frame buffer locking is being enabled in the example below.:

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

Remote Testing

The frame buffer locking option does not work when you start `sunvts` or `vtsk` remotely. In this case, set the frame buffer locking option to disable. Do not run any graphic programs (including vtsui) on that frame buffer during graphic testing.

1

`audio` *Test*

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

The audio device is an *exclusive use device*. Only one process or application can interface with it at a time. This test is not scalable.

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

`audio` *Subtests*

Record/Play Test

This test plays and records one second of data. No data checking is done. 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. 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, the music plays for about 30 seconds, then stops, and returns an error. 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 user must decide if the test passes or fails. Badly distorted audio, or no audible music indicates a problem. This test is supported on all audio implementations.

`audio` *Options*

Upon start-up, the SunVTS probe utility determines which audio implementation is present and adjusts the audio option menu appropriately. The possible option menus are as follows:

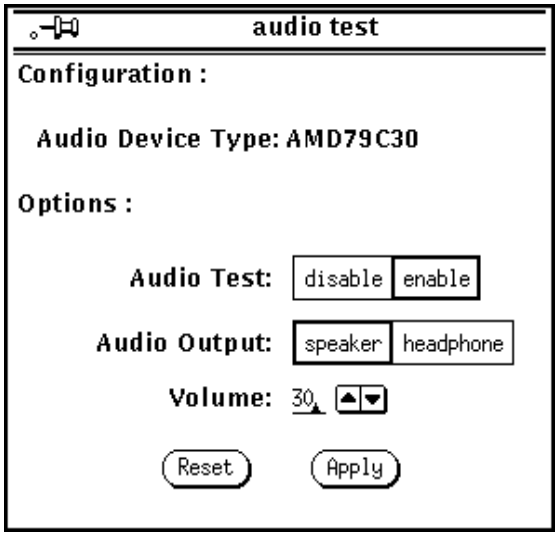


Figure 2-1 `audio` Test Option Menu for `audioamd(7)`

The `audioamd(7)` is an 8-bit mono, telephone-quality, audio device.

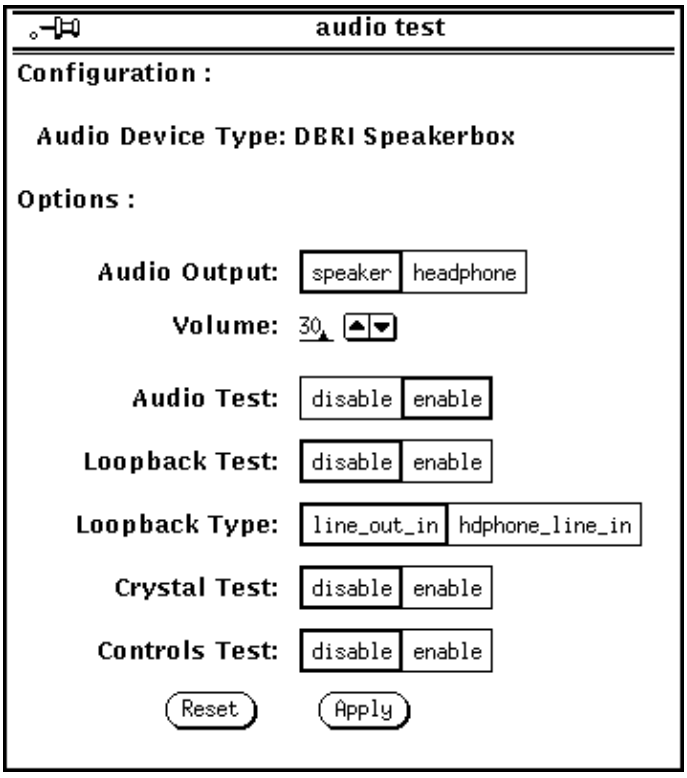


Figure 2-2 `audio` Test Option Menu for speakerbox `dbri(7)` `audio`

The speakerbox `dbri` is a high-quality 16-bit stereo multimedia codec.

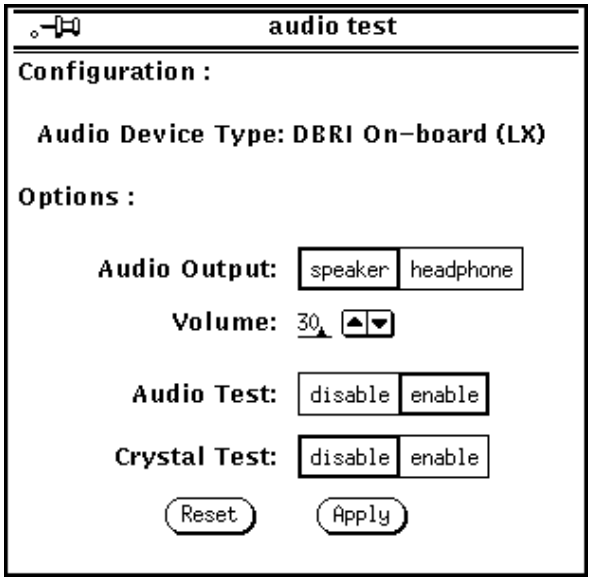


Figure 2-3 `audio` Test Option Menu for SPARCstation LX `dbri(7)` with no speakerbox attached

This is an on-board, high-quality 16-bit stereo multimedia codec.

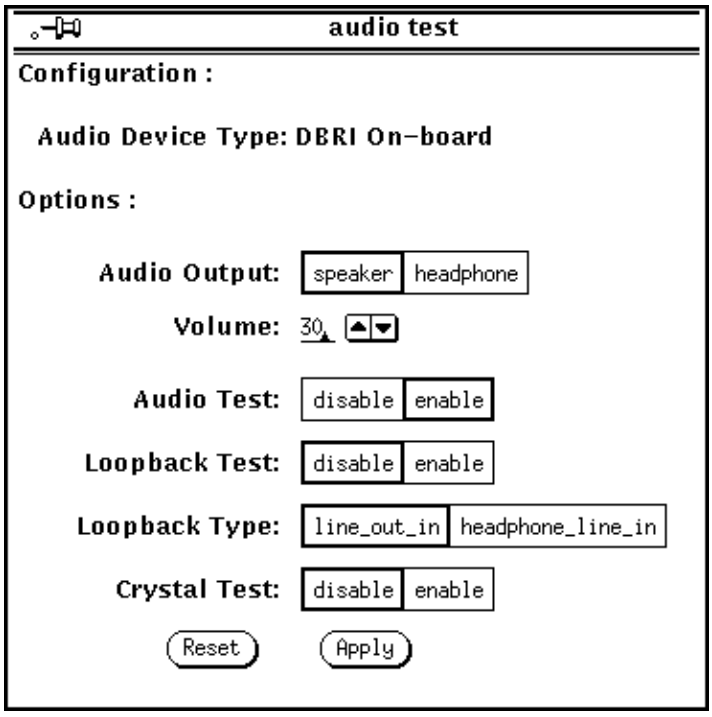


Figure 2-4 `audio` Test Option Menu for on-board `dbri(7)` for SPARCstation 20 and S240

This is an on-board, high-quality 16-bit stereo multimedia codec.

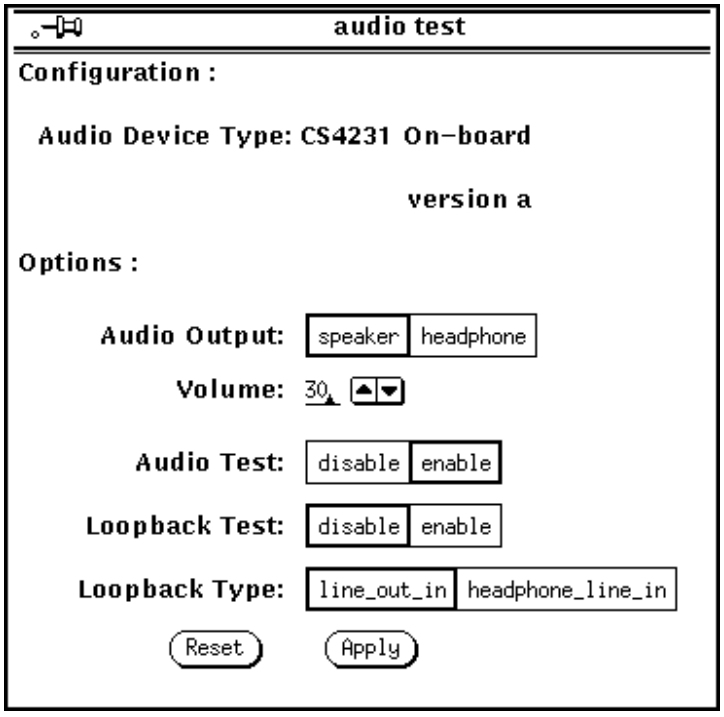


Figure 2-5 `audio` Test Option Menu for `audiocs(7)`

The `audiocs(7)`is an on-board, high-quality 16-bit stereo multimedia codec.

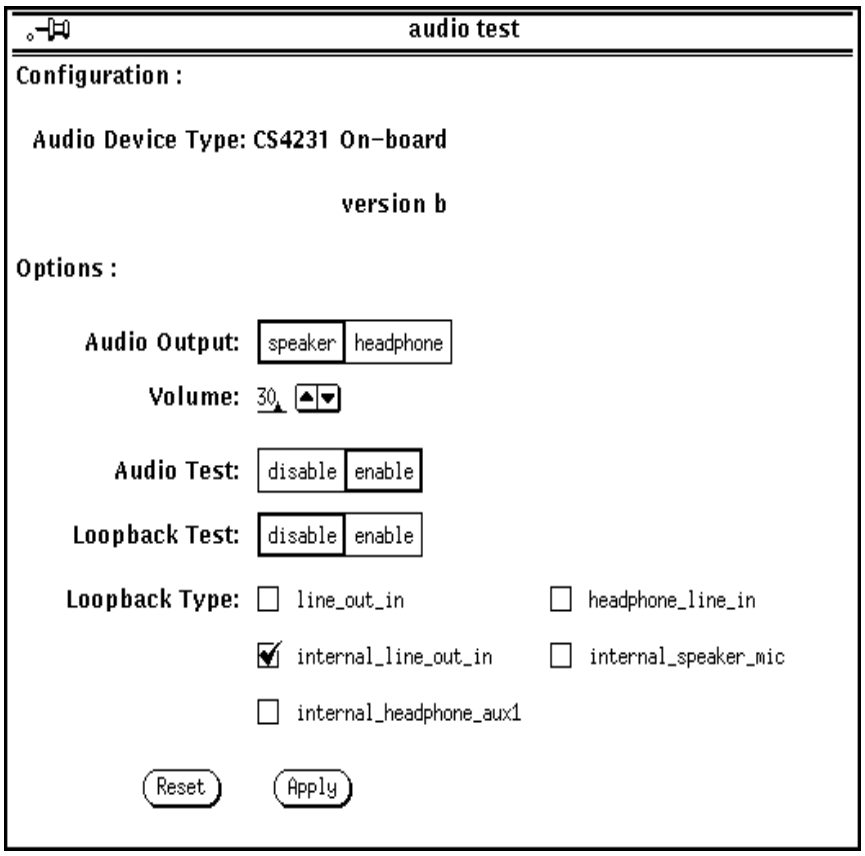


Figure 2-6 `audio` Test Option Menu for `audiocs`(7) with Internal Loopbacks

The `audiocs`(7) is an on-board, high-quality 16-bit stereo multimedia codec.

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 “audio Command Line Syntax” on page 18.

- ***Audio Test*** enables or disables the music play test. This test is enabled by default.

- ***Audio Output*** selects the output port for the music play test.
- ***Volume*** sets the volume for the music play test.
- ***Loopback Test*** enables or disables the loopback test. A loopback cable must be installed between the selected ports to run this test. 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 and 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.

`audio` *Test Modes*

Connectivity Mode

In connectivity mode a simple open and close is done on the audio device. 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.

Online Mode

In online mode the record / play test is run. 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.

Offline Mode

In offline mode the record / play test is run and the user can optionally run any of the tests described earlier. In offline mode, the test will fail if the device is busy.

2

audio *Command Line Syntax*

/opt/SUNWvts/bin/audio *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*

Argument	Explanation
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.
T=*loopback_test_type*	Specifies the type of loopback test. The default is **1**; the choices are listed below. **0** Speaker / Microphone **1** Line-in / Line-out **2** Headphone / Line-in **3** Headphone / Microphone **4** Speaker / CD-input **I1** *I*nternal Line-in / Line-out **I2** Internal Spk / Mic **I3** Internal Headphone / Aux1 Note: Test types 0, 3, an 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.

Argument	Explanation *(Continued)*
CD	Enables the `cdtest`. This is for systems with an internal CD-ROM drive. A CD disc 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 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.

2

audio *Error Messages*

The audio error messages are listed below with explanations or probable causes:

```
6000 Signal To Noise ratio too low (<name>) on <name>, SNR
=<number> db, Min SNR =<number> db (<text>)
```

Probable_Cause(s):

a. **Loopback cable is missing or faulty**

b. **Audio hardware (usually consistent failures)**

c. **System software problem (usually intermittent failures)**

```
6001 Channel Separation too low (<name>), SEP =<number> db,
Min SEP =<number> db (<text>)
```

Probable_Cause(s):

a. **Loopback cable is faulty**

b. **Audio hardware (usually consistent failures)**

c. **System software problem (usually intermittent failures)**

```
6002 Loop gain is out of range (<name>), <name> GAIN
=<number> db, Min =<number> Max =<number> db (<text>)
```

Probable_Cause(s):

a. **Loopback cable is missing or faulty**

b. **Audio hardware (usually consistent failures)**

c. **System software problem (usually intermittent failures)**

```
8000 Must be super user (root) to execute
```

The user does not have root privileges.

Probable_Cause(s):

a. **User does not have root privileges**

8012 Invalid audio device (<device_name>) for Crystal test

Probable_Cause(s):

a. The crystal test is not supported on the system audio device.

8013 Invalid audio device (<device_name>) for Controls test

Probable_Cause(s):

a. The system audio device does not support a speakerbox.

8014 Invalid audio device (<device_name>) for Loopback Quality test

Probable_Cause(s):

a. No loopback tests are supported on the system audio device.

Probable_Cause(s):

a. Loopback testing is not supported on this audio device

8015 Invalid audio device (<device_name>) for Loopback Quality test (T=<number>)

Probable_Cause(s):

a. The specified loopback test is not supported on the system audio device.

b. The specified loopback type is not supported on this audio device

8020 Incomplete button press sequence

Probable_Cause(s):

a. The required button presses were not detected in the time allowed.

```
8023 Underrun/Overrun error failure
```

Probable_Cause(s):

a. The audio driver detected underrun or overrun errors while transferring data to the audio device. These error usually happen when the loopback test is run while there is heavy system activity.

```
8027 <name>: '<name>(<name>=<number>, <name>=<number>,
<name>=<number>)' system call timeout. No response after
<number> seconds. Device = <device_name>
```

Probable_Cause(s):

a. The system call is hanging.

b. Audio hardware (usually consistent failures)

c. System software problem (usually intermittent failures)

```
8028 Audio crystal test did not complete in the expected
time, rate = <number> Hz, time = <number> usecs, min =
<number>, max = <number>
```

Probable_Cause(s):

a. Excessive loading on the system

Recommended_Action(s):

a. Disable all other tests and try again

```
8029 Speaker is an invalid output port for device
```

Probable_Cause(s):

a. The system does not have an on-board speaker.

```
8032 Failed ioctl <name> (<error_message>)
```

Probable_Cause(s):

a. The `ioctl`( ) system call failed. See the appended error message.

8033 Failed mmap (<error_message>)

Probable_Cause(s):

a. The `mmap`( ) system call failed. See the appended error message.

8034 Failed fstat (<error_message>)

8035 Failed ftruncate (<error_message>)

Probable_Cause(s):

a. The `ftruncate`( ) system call failed. See the appended error message.

8036 Write to audio device returned error condition (<error_message>)

Probable_Cause(s):

a. The `write`( ) system call failed. See the appended error message.

8037 Read from audio device returned error condition (<error_message>)

Probable_Cause(s):

a. The `read`( ) system call failed. See the appended error message.

8038 Failed to allocate <number> bytes of memory (<error_message>)

Probable_Cause(s):

a. The memory allocation system call failed, most likely due to a lack of system memory resource. See the appended error message.

8051 Unknown audio device (name=<device_name>, config=<name>).

8052 Unable to get platform name.

8053 Unknown architecture/audio = <name> / <device_name>.

8055 Could not open <device_name>: <error_message>

The open( ) system call failed. See the appended error message.

Probable_Cause(s):

a. device does not respond

b. no mmcodec device found

c. Unable to communicate with speakerbox

Recommended_Action(s):

a. Examine system message files (/var/adm/messages) for other information

8075 Invalid audio file format

Probable_Cause(s):

a. Specified audio file is not in expected format.

8077 Invalid audio encoding

Probable_Cause(s):

a. An unknown audio encoding type was specified.

8090 Only one <name> loopback can be selected at a time

Probable_Cause(s):

a. The user specified more than one loopback test on the same port.

8091 Cannot enable loopback testing without selecting a loopback type

Probable_Cause(s):

a. The user enabled loopback testing but did not specify a loopback type.

bpptest 3

*Bidirectional Parallel Port Printer Test (*bpptest*)*

The bidirectional parallel port printer test (`bpptest`) verifies the functionality of the bidirectional parallel port. SBus printer cards have two printer ports: one for SPARCprinters™, and one for any parallel port printer. Use the `lpvitest` to verify SPARCprinter functionality.

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 occur if 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 this test 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.

3

Printer Test Hardware and Software Requirements

The SBus printer card and device drivers must be installed to run `lpvitest` or `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 hooked up to the SBus card, you can test both 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*

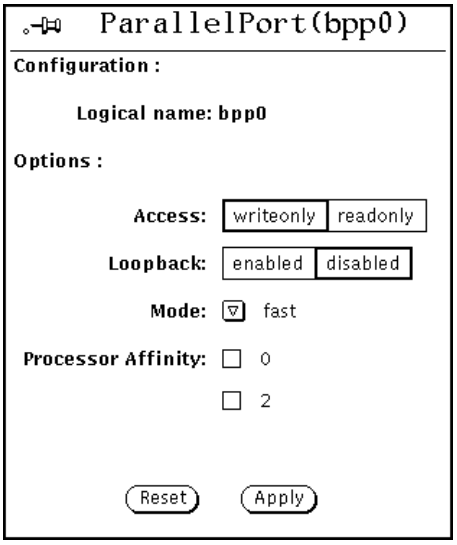


Figure 3-1 `bpptest` Option Menu

- ***Access*** determines the direction of data transfer; this field is informational only. `writeonly` is the only option currently available. This indicates that the only data being transferred is going 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 offline modes this option is set to medium. In the offline mode, the choices are:
  - `Fast` Prints an image every 10 seconds.
  - `Mediu` 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 all the three testing modes, namely connectivity, online and offline.

Connectivity Mode

This mode verifies that a bidirectional parallel port is configured on the system. The success of the bpptest in this mode indicates that the system has the bidirectional parallel port hardware and the software driver is installed.

Online Mode

In this mode the test attempts to verify the functionality of the parallel port and the path between the parallel port and the host memory. An internal loopback test is performed, and if a printer is attached to the port a data pattern is transferred to the printer.

If the port is found to be busy at the time of testing, a message is printed on the SunVTS console indicating that the device is unavailable for testing and the error and pass counts are unaffected.

Offline Mode

The testing done in this mode is the same as in the online mode, except that the test 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 offline mode 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*`,`**`loop=`***disable* | *enable*

Argument	Explanation
`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.

`bpptest` *Error Messages*

The `bpptest` error messages are listed below with explanations or probable cause.

```
6000 Read compare failed, expected:<value>,
observed:<value>
```

Probable_Cause(s):

a. Suspect DMA circuitry or FIFO associated with bpp

6001 Error occurred while accessing <device_name>; Time out Error: <value>; Bus Error: <value>; Pin Status: <value>

Probable_Cause(s):

a. <system_error_message>

8000 could not open <device_name>

Probable_Cause(s):

a. <system_error_message>

b. Device does not exist

c. Device not configured correctly

8001 Encountered error while writing to <device_name>; Requested bytes to write: <value>; Bytes successfully written: <value>

Probable_Cause(s):

a. Device not configured correctly or does not exist

b. Too many print requests queued up

c. Printer out of paper

8002 Failed to obtain memory from malloc() system call

Probable_Cause(s):

a. System heavily loaded

8003 Error occurred while attempting to fork a child process

Probable_Cause(s):

a. <system_error_message>

```
8004 ioctl fail on <device_name>
```

Probable_Cause(s):

a. <system_error_message>

```
8005 error=<number>
```

Probable_Cause(s):

a. <system_error_message>

```
8006 Device <device_name> unavailable for testing
```

Probable_Cause(s):

a. Device is busy

Recommended_Action(s):

a. Retry later

cdtest 4

*Compact Disc Test (*cdtest*)*

This test 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 scratch compact disc into the drive before starting the test. See the explanation for CD_type in the chart in Section , “cdtest Test Modes.”

Volume Management and Compact Discs

The `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), then 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.

4

`cdtest` *Options*

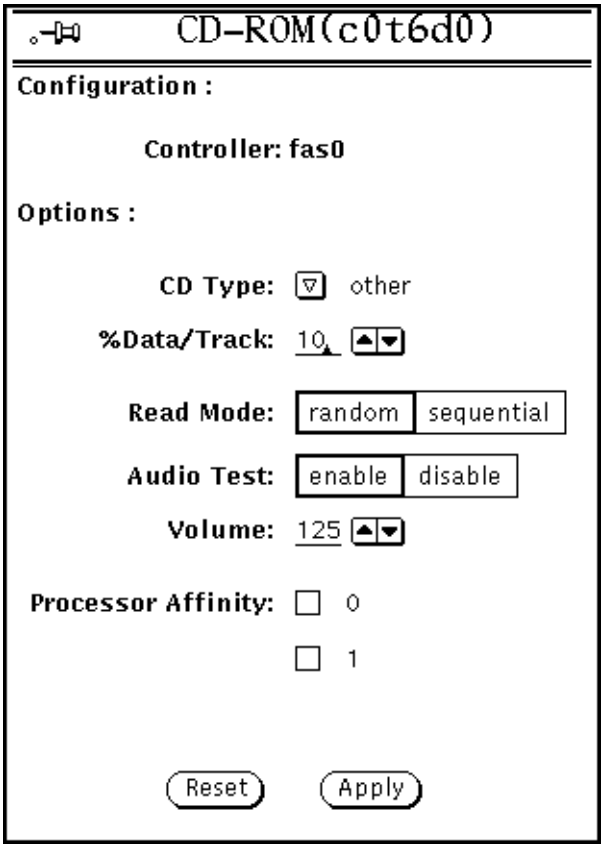


Figure 4-1 `cdtest` Option Menu

- ***CD Type*** indicates the type of compact disc to test from the CD Type menu. The choices are: *pdo*, *multi-session*, or *other* (the default CD type is *other)*.In the connectivity mode this option has a canned 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 connectivity modes this option has a canned value of 2%.

- ***Read Mode*** indicates the mode; choose between Random or Sequential reading. 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 connectivity modes 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 connectivity mode this option has a canned value of **disable.**
- ***Volume*** adjusts the volume; type a value between 0 and 255 in this field. In the online and connectivity modes this option has a canned value of 125.

`cdtest` *Test Modes*

This test supports all the three modes of testing available under SunVTS, namely connectivity, online and offline.

Connectivity Mode

In this mode the test verifies that a CD-ROM drive is connected to and configured in the system.

Online Mode

This mode attempts to verify the functionality of the drive by accessing and reading the media in the drive. If the media contains audio and the audio test is enabled, it tries to play it.

Only a small percentage of the media is read, as this is sufficient to verify the functionality of the drive and keeps the test runtime from being unnecessarily extended.

If the device is found to be busy at the time of testing the test exits after printing a message indicating that the device is unavailable for testing. The pass and error counts are unaffected.

Offline Mode

This mode is similar to the 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 offline mode and therefore the unavailability of the 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*

Argument	Explanation
`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`** through **`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 **`others`**; the default is **other**.

`cdtest` *Error Messages*

6000 Fail to seek at block <value>

Probable_Cause(s):

a. <system_error_message>

b. Bad media

Recommended_Action(s):

a. Retry with a different CD-ROM

6001 Fail to read <number> bytes at block <number>

Probable_Cause(s):

a. <system_error_message>

b. Bad media

c. Incorrect CD-ROM type passed to cdtest

6002 Data mode <value> not supported

Probable_Cause(s):

a. datamode has a value other than CD_DATAMODE1 or CD_DATAMODE2 (see cdio(7))

6003 Error occurred while playing audio track <number>

6004 Audio stop before complete at track <number>

6005 Unexpected audio status <value> received at track <number>

6006 Fail in reading offset of the multi-session, offset = <value>

Probable_Cause(s):

a. An incorrect value for cdtype has been passed to the cdtest

b. Bad media

6007 This is not a CD ROM Multi-Session disc!

Probable_Cause(s):

a. An incorrect value for cdtype has been passed to the cdtest

8000 Fail to open device <device_name>

Probable_Cause(s):

a. <system_error_message>

8001 Fail ioctl <name>

Probable_Cause(s):

a. <system_error_message>

8002 Fail ioctl <name>

Probable_Cause(s):

a. <system_error_message>

b. No partitions on the disc

8003 Fail to allocate <number> bytes of memory

8004 Error occurred while reading TOC header

Probable_Cause(s):

a. Track numbers out of range; First track: <value>, Last track: <value>

Recommended_Action(s):

a. Retry with different CD-ROM

8005 Data miscompare while reading <number> bytes starting from block <number>

Probable_Cause(s):

a. An incorrect value for cdtype has been passed to cdtest

b. Bad media

```
 8006  ioctl CDROMSTART failure (number of retries:
<value>)
```

4

cg14test 5

*Color Graphics Frame Buffer Test (*cg14test*)*

This test checks the cg14 frame buffer card. The 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:

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

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.

Ensure that the frame buffer locking option is enabled from the Options window (see the section about testing frame buffers in *SunVTS 2.0 User's Guide* for details).

5

`cg14test` *Groups*

There are nine test groups with cg14test:

- MDI and VBC Chip Control Registers
- Memory Chips
- MDI Chip Cursor Registers
- MDI Chip CLUT Registers
- DAC Chip Registers
- MDI Chip XLU Registers
- CG14 Display (visual only)
- MDI Chip Testmode Readback in 8-bit mode
- Driver IOCTLs

Hardware Test Groups (test groups 1 - 6)

Testing is done by opening `/dev/fbs/cgfourteenX`, mmap'ing (R/W Shared) the MDI Control Address Space, modifying the target test location (using direct writes to the mmap'ed address space), reading from the mmap'ed address space for verification, and closing the device.

Visual Pattern Test Group (test group 7)

Testing is done by loading a visual pattern of 256 colors, then rotating the pattern around by adjusting CLUT1. This subtest must be verified visually.

Data Propagation Test Group (test group 8)

Testing is done 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 permit, 8-bits per pixel mode are tested.

Driver Test Group (test 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, as follows:

1. Saves register data before it is overwritten.
2. Disables video, if possible.
3. Performs the specific test.
4. Restores 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 under test.

MDI and VBC Chip Control Registers (test 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 (test group 2)

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 5-1).

The test operates as follows:

- It ascends through the FB 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.

Table 5-1 `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

Table 5-1 `cg14test` NTA Testing Patterns

NTA Test Pattern Number = *x*	**Test Data** = *y*	**New Data** = *z*
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	-

Memory Retention

VRAM Data Retention 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.

There are two new command line parameters 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.

Test 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 being 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

MDI Chip Cursor Registers (test group 3)

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 (test group 4)

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).

DAC Chip Registers (test group 5)

This section describes 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 (test group 6)

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 (withauto increment).

CG14 Display (visual only) (test group 7)

This 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 [TMRB] (test group 8)

Test Mode Readback Register bits 23-0 read-only verify.

Driver IOCTLs (test 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 use!).
- 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.

`cg14test` *Options*

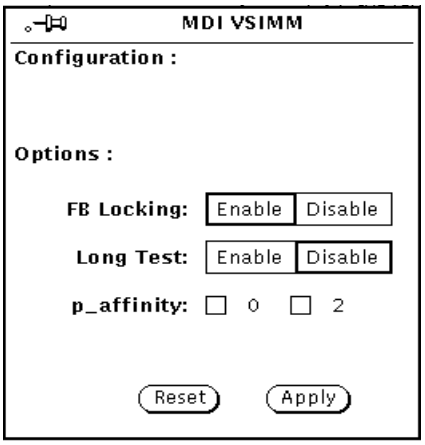


Figure 5-1 `cg14test` Option Menu

- ***FB Locking*** — See the section about Testing Multiple Frame Buffers in the *SunVTS 2.0 User's Guide* 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 first SAM transfer only.
- ***p_affinity*** — For multiprocessor systems, indicates the processor to be tested.

5

`cg14test` *Command Line Syntax*

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

Argument	Explanation
`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 the section about Testing Multiple Frame Buffers in *SunVTS 2.0 User's Guide* for details. Do not use when 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.

`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 mode.

Connectivity Mode

Not applicable for this test.

Online Mode

Not applicable for this test.

Offline Mode

The offline mode uses all subtests to test the cg14 frame buffer. The user can select the long mode for TRMB subtest.

`cg14test` *Error Messages*

```
6002 MDI <name> register = <number>
```

Probable_Cause(s):

a. CG14 video board

```
6004 MDI Cursor Planes Retry test maximum retry limit
exceeded
```

Probable_Cause(s):

a. CG14 video board

```
6006 <name> error message
```

Probable_Cause(s):

a. CG14 video board

```
6008 failed open of file <name>, errno=<number>
```

Probable_Cause(s):

a. Disk

b. CPU board

```
6010 failed write to file <name>, errno=<number>
```

Probable_Cause(s):

a. CG14 video board

b. CPU board

c. Disk

```
6012 failed close of file <name>, errno=<number>
```

Probable_Cause(s):

a. Disk

b. CPU board

```
6014 MDI chip TestMode Readback, <number>-bit <name> mode,
offset= <number> pixelpipe=<name> clock=<number>
exp=<number> obs=<number>

6016 MEM (<name>), Data Retention offset= <number>
exp=<number> obs=<number>
```

Probable_Cause(s):

a. CG14 video board

b. CPU board

```
6018 MEM (<name>), NTA <name> offset= <number> exp=<number>
obs=<number>
```

Probable_Cause(s):

a. CG14 video board

b. CPU board

```
6020 MEM (<name>), WRRD <name> offset= <number>
exp=<number> obs=<number>
```

Probable_Cause(s):

a. **CG14 video board**

b. **CPU board**

```
6022 failed, mapping <name> space, errno = <number>
```

Probable_Cause(s):

a. **CG14 device file**

b. **SunOS**

c. **CG14 video board**

d. **CPU board**

```
6024 MEM (<name>), WRRD <name> offset= <number>
exp=<number> obs=<number>
```

Probable_Cause(s):

a. **SunOS**

b. **CG14 video board**

c. **CPU board**

```
6026 VBC Control Register exp=<number> obs=<number>
```

Probable_Cause(s):

a. **CG14 video board**

b. **CPU board**

```
8002 <name> is an invalid parameter for <name>!
```

Probable_Cause(s):

a. **Operator error**

```
8004 unable to close device <name>
```

Probable_Cause(s):

a. CG14 device file

b. SunOS

c. CG14 video board

```
8006 invalid CG14 device type from <name>
```

Probable_Cause(s):

a. CG14 device file

b. SunOS

c. CG14 video board

```
8008 IOCTL Error: <name>
```

Probable_Cause(s):

a. CPU board

b. CG14 video board

```
8008 IOCTL(<name>) <name>
```

Probable_Cause(s):

a. CPU board

b. CG14 video board

```
8010 unable to unlock OL windows
```

Probable_Cause(s):

a. SunOS

b. Operator error

```
8012 IOCTL(<name>,CLUT<number>) <name>
```

Probable_Cause(s):

a. CG14 device file

b. SunOS

c. CG14 video board

```
8014 unable to open CG14 device <name>
```

Probable_Cause(s):

a. Incorrect device name

b. No existing device

c. CG14 video board

5

cg6 6

*Frame Buffer, GX, GX+ and TGX Options Test (*cg6*)*

The cg6 test verifies the cgsix 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.

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

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

- ***Cursor Test*** visually checks the overlay registers of the RAMDAC. A pointer is drawn on the screen and moved around 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. Different patterns are written to each of them. 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 being accessed. This confirms that further TEC access is performed correctly.
- ***FBC Test*** verifies that the Frame Buffer Controller logic is being accessed. This confirms that further FBC access is performed correctly.
- ***Frame Buffer Test*** verifies that the frame buffer memory is working. A walking 1 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 put 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.
- ***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 backwards and forwards. 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*

FB Locking

See the section about Testing Frame Buffers in the *SunVTS 2.0 User's Guide* for details about frame buffer locking.

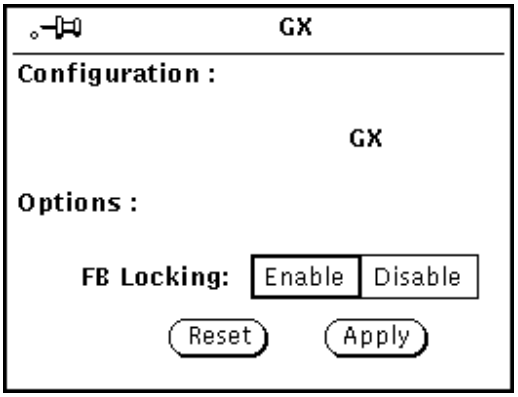


Figure 6-1 cg6 Options Menu

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. This test is only available in offline mode.

Connectivity Mode

Not applicable for this test.

Online Mode

Not applicable for this test.

Offline Mode

The cg6 test verifies the cgsix frame buffer (GX, GX+ or TGX) offered with most SPARC TM based workstations and servers. The subtests are described in the first section of this chapter.

6

cg6 *Command Line Syntax*

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

Argument	Explanation
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 the section about Testing Multiple Frame Buffers in *SunVTS 2.0 User's Guide* for details. Do not use when device is the window system display.
L	Enables the long TMRB test.
Passes=*number*	Indicates the number of passes to run; the default is 1 pass.
I	Enables optional driver ioctl tests for cursor. `Note:` Do not move the mouse during the `cg14test` when you run this option.

Note – Extra swap space of 5 MB is required.

cg6 *Error Messages*

The error messages described below are returned by `cg6` for subtest failures:

```
6000 Resolution not supported for double buffering.
```

Recommended_Action(s):

a. Change to other resolutions.

```
6002 data error: x_pos: <number>, y_pos: <number>. *mobs:
<number>, *mexp: <number>.
```

Probable_Cause(s):

a. Faulty Frame Buffer

Recommended_Action(s):

a. If the problem persists, call your authorized Sun service provider.

```
6004 FB Screen Test data error:  xpos=<number>,
ypos=<number>, exp(<number>), obs(<number>)
```

Probable_Cause(s):

a. Faulty Frame Buffer

Recommended_Action(s):

a. If the problem persists, call your authorized Sun service provider.

```
6006 Data Error: Screen location x: <number>, y: <number>,
Obs: <number>, Exp: <number>
```

Probable_Cause(s):

a. Faulty Frame Buffer

Recommended_Action(s):

a. If the problem persists, call your authorized Sun service provider.

```
6007 Multibuffering extension does not exists.
```

Probable_Cause(s):

a. Faulty Frame Buffer

Recommended_Action(s):

a. If the problem persists, call your authorized Sun service provider.

```
6008 Cannot create CMAP
```

Probable_Cause(s):

a. System error

b. Faulty Frame Buffer

Recommended_Action(s):

a. If the problem persists, call your authorized Sun service provider.

```
6009 <name>
```

off(<number>)exp(<number>), obs(<number>), xor(<number>)

Probable_Cause(s):

a. Faulty Frame Buffer

Recommended_Action(s):

a. If the problem persists, call your authorized Sun service provider.

```
6010 Could not get color
```

Probable_Cause(s):

a. System error

b. Faulty Frame Buffer

Recommended_Action(s):

a. If the problem persists, call your authorized Sun service provider.

```
6011 <name>
```

off(<number>)exp(<number>), obs(<number>), xor(<number>) <name>

Probable_Cause(s):

a. Faulty Frame Buffer

Recommended_Action(s):

a. If the problem persists, call your authorized Sun service provider.

```
6012 Checksum test failed for PROM. Observed: <number>,
Expected: <number>
```

Probable_Cause(s):

a. Faulty Frame Buffer

Recommended_Action(s):

a. If the problem persists, call your authorized Sun service provider.

```
6014 Unable to lock frame buffer.
```

Probable_Cause(s):

a. FB lock was enabled while Window System was not running.

```
6016 Found error in sine test
```

Probable_Cause(s):

a. Faulty Frame Buffer

Recommended_Action(s):

a. If the problem persists, call your authorized Sun service provider.

```
6018 Found error in cursor test
```

Probable_Cause(s):

a. Faulty Frame Buffer

Recommended_Action(s):

a. If the problem persists, call your authorized Sun service provider.

```
6020 Found error in fb test
```

Probable_Cause(s):

a. Faulty Frame Buffer

Recommended_Action(s):

a. If the problem persists, call your authorized Sun service provider.

```
6022 Found error in polygon test
```

Probable_Cause(s):

a. Faulty Frame Buffer

Recommended_Action(s):

a. If the problem persists, call your authorized Sun service provider.

```
6024 Found error in cmap test
```

Probable_Cause(s):

a. Faulty Frame Buffer

Recommended_Action(s):

a. If the problem persists, call your authorized Sun service provider.

```
6026 Found error in db test
```

Probable_Cause(s):

a. Faulty Frame Buffer

Recommended_Action(s):

a. If the problem persists, call your authorized Sun service provider.

```
6028 Found error in line test
```

Probable_Cause(s):

a. Faulty Frame Buffer

Recommended_Action(s):

a. If the problem persists, call your authorized Sun service provider.

```
6030 Found error in dac test
```

Probable_Cause(s):

a. Faulty Frame Buffer

Recommended_Action(s):

a. If the problem persists, call your authorized Sun service provider.

8000 <name> not mapped.

Probable_Cause(s):

a. System error

Recommended_Action(s):

a. If the problem persists, call your authorized Sun service provider.

8002 can’t acquire console semaphore

Probable_Cause(s):

a. System error

Recommended_Action(s):

a. If the problem persists, call your authorized Sun service provider.

8004 XCreateSimpleWindow failed

Probable_Cause(s):

a. System error

Recommended_Action(s):

a. If the problem persists, call your authorized Sun service provider.

8008 Couldn’t create enough buffers.

Probable_Cause(s):

a. Not enough memory available

b. System error

Recommended_Action(s):

a. Reduce system load

b. If the problem persists, call your authorized Sun service provider.

8010 grab of pointer unsuccessful

Probable_Cause(s):

a. Not enough memory available

b. System error

Recommended_Action(s):

a. Reduce system load

b. If the problem persists, call your authorized Sun service provider.

```
8012 cmap mapping failed.
```

Probable_Cause(s):

a. Not enough memory available

b. System error

Recommended_Action(s):

a. Reduce system load

b. If the problem persists, call your authorized Sun service provider.

```
8014 Blit status
```

Could not launch draw

Probable_Cause(s):

a. Faulty Frame Buffer

Recommended_Action(s):

a. If the problem persists, call your authorized Sun service provider.

```
8016 Draw Wait FBC_DRAW status=<number>
```

Probable_Cause(s):

a. Faulty Frame Buffer

Recommended_Action(s):

a. If the problem persists, call your authorized Sun service provider.

8028 Could not get cmap

Probable_Cause(s):

a. Faulty Frame Buffer

Recommended_Action(s):

a. If the problem persists, call your authorized Sun service provider.

8020 Could not get color

Probable_Cause(s):

a. Faulty Frame Buffer

Recommended_Action(s):

a. If the problem persists, call your authorized Sun service provider.

8022 Could not create child raster.

Probable_Cause(s):

a. Incorrect device name

b. Not enough memory available

Recommended_Action(s):

a. Re-enter the correct name

b. Reduce system load.

8024 Failed to get malloc

Probable_Cause(s):

a. Not enough memory available

b. System error

Recommended_Action(s):

a. Reduce system load.

```
8026 Cannot create CMAP
```

Probable_Cause(s):

a. System error

b. Faulty Frame Buffer

Recommended_Action(s):

a. If the problem persists, call your authorized Sun service provider.

```
8028 Couldn't get hostname of machine under test.
```

Probable_Cause(s):

a. System error

Recommended_Action(s):

a. If the problem persists, call your authorized Sun service provider.

```
8030 Failed in opening device <name>
```

Probable_Cause(s):

a. Incorrect device name

b. Faulty Frame Buffer

Recommended_Action(s):

a. Re-Enter the correct name

b. If the problem persists, call your authorized Sun service provider.

```
8032 Failed to create raster.
```

Probable_Cause(s):

a. Incorrect device name

b. Not enough memory available

Recommended_Action(s):

a. Re-Enter the correct name

b. Reduce system load.

```
8034 Failed to create context.
```

Probable_Cause(s):

a. Not enough memory available

b. System error

Recommended_Action(s):

a. Reduce system load

b. If the problem persists, call your authorized Sun service provider.

6

disktest 7

The `disktest` verifies the functionality of hard disk drives and floppy drives using two subtests. 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 on the top of the option menu. The 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. The File System subtest verifies the disk system's integrity.

Initially, `disktest` probes the disks under `/dev/rdsk`. It checks all of the partitions of each disk. If any partition has a file system not yet mounted, `disktest` pre-mounts these partitions for the File System subtest. The pre-mount point bears the name of the disk partition appended with a system-wide unique number. For example, if the disk name is `/dev/dsk/c0t3d0`, `disktest` mounts it as superuser under the name `/c0t3d0.XXXXXX`. Where XXXXXX is a six-digit system-wide number. The option menu shows all partitions available for testing. The File System subtest can only be run if the selected partition *is* mounted. The Write option of the Media subtest is allowed only if selected partition is *not* mounted.

The `disktest` tests the floppy drive regardless of whether or not the Volume Management software is running.

- If the Volume Management software is running, `disktest` tests the disk drive with the mount point name in the `/etc/mnttab` file.

Caution – If a power failure occurs while the Media subtest is being run in write mode, disk data will be destroyed.

- 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 floppy drives. Currently, the sunvts software is hard-coded to use these path names as the default logic name.

The `disktest` performs a random seek check followed by a read test or a read after write test. The Media subtest is a scalable test; so you can run multiple copies of it 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 File System subtest exercises the partition being tested determine if it is mounted. If the partition is not already mounted or pre-mounted, then the test is blocked.

`disktest` *Test Options*

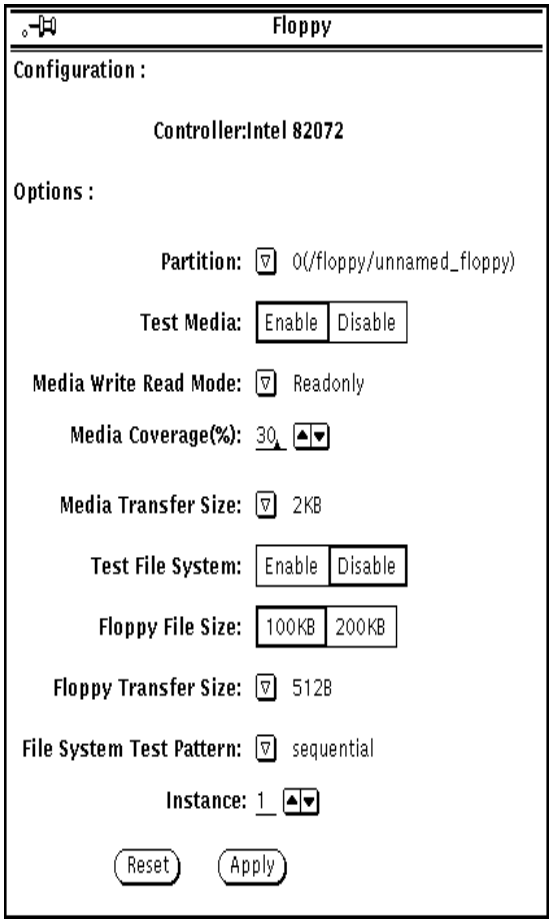


Figure 7-1 `disktest` Configuration and Options Menu

The `disktest` has different option menus for different test modes.

Partition: The partitions for the Media subtest are displayed. If a partition is mounted, it's 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: Enable or Disable of Media subtest.

Mode: Read only or Rad after write with / without backup.

Media Coverage(%): Percentages of partition will be tested.

Media Transfer Size: Transfer size of Media subtest.

Test File System: Enable or Disable of File System subtest File System File Size: Size of files created for testing.

File System Transfer Size: Transfer size of File System subtest File System Test Pattern: Test pattern of File System subtest.

Connectivity Mode

Option Menu for hard disk partition: 0 - 7 [default]

Test Media: [Enable~]	(fixed to Enable)
Test Mode: [ReadOnly~]	(fixed to Readonly)
Media Coverage(%): [1]	(default - can be changed)
Media Transfer Size: [2KB]	
Test File System: [Disable~]	(fixed to Disable)

Online Mode

On-line Mode for hard disk partition: 0 - 7 [default]

Test Media: [Enable]<- [Disable]	
Test Mode: [Readonly~]	(fixed to Readonly)
Media Coverage(%): [10]	
Media Transfer Size: [2KB]	[10KB] [20KB] <- [40KB] [60KB]
Test File System: [Disable~]	(fixed to Disable)

Offline Mode for hard disk (under SCSI-devices group): partition: 0 - 7 [default]

Test Media: [Enable]<- [Disable]

Mode: [ReadOnly]<- [BackupWriteRead] [NoBackupWriteRead]

Media Coverage(%): [30]

Media Transfer Size: [2KB] [10KB] [20KB] <- [40KB] [60KB]

Test File System: [Enable] [Disable] <-

File System File Size:[512KB]<- [2MB] [8MB] [20MB] [100MB] [200MB]

File System Transfer Size:[512B]<- [1024B] [10KB] [40KB] [100KB]

File System Test Pattern:[sequential] [0x00000000] [0xffffffff] [0x5aa55aa5] [0xdb6db6db] [random]

Offline Mode for floppy disk (under Other-Devices group): partition: 0 - 7 [default]

Test Media: [Enable]<- [Disable]

Mode: [ReadOnly]<- [BackupWriteRead] [NoBackupWriteRead]

Media Coverage(%): [30]

Media Transfer Size: [2KB]<- [10KB] [20KB]

Test File System: [Enable] [Disable] <-

File System File Size:[512KB]<- [2MB] [8MB] [20MB] [100MB] [200MB]

File System Transfer Size:[512B]<- [1024B] [10KB]

File System Test Pattern:[sequential] [0x00000000] [0xffffffff] [0x5aa55aa5] [0xdb6db6db] [random]

Note – Only the offline mode test is allowed on floppy `disktest`.

7

`disktest` *Test Modes*

The `disktest` supports all three modes. It performs different test schemes on the network device, according to the mode you select.

Connectivity Mode

This mode is only available for the hard disk test. There is no Connectivity Mode for floppy driver test. Only one instance of disktest is allowed for each disk device. A `disktest` monitors UNIX error messages. The 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 will be run. No Write option is available in Connectivity Mode.

Online Mode

The Online Mode is only available for the hard disk test. There is no Online Mode for the floppy driver test. Only one instance of `disktest` is allowed for each disk device. A `disktest` monitors UNIX error messages. `disktest` displays messages and reports errors.

In this mode, `disktest` also opens the hard disk, checks the disk configuration, and executes the Media subtest. Next, `disktest` performs some random seek checks. Only Read Only mode is allowed. The default coverage is 10%. No File System subtest will be run. When the test finishes, `disktest` closes the disk device being tested. No Write option is available in Online Mode.

Offline Mode

More than one instance of `disktest` is allowed for one disk device. Both File system subtest and Media subtest can be run in Offline mode. The `disktest` does not monitor UNIX error messages in this mode. Floppy test can be run in Off-line mode.

`disktest` *Command Line Syntax*

`/opt/SUNWvts/bin/disktest` *standard_arguments* **`-o p=n,-i=n, -w=n,dev=`**<*device_name*>**`,partition=<0-7>,rawsub=E/D,rawrw=, rawcover=,rawiosize=,fssub=,fssize=,fsiosize=,fspattern=,`**

Argument	Explanation
`p=`*number*	Sets the number of passes for the test to run; default is 1.
`i=`*number*	Sets the number of total instances for the test; default is 1.
`w=`*number*	Determines which instance this test is assigned; default is 0.
`dev=`<*device_name*>	Specifies the name of the disk to be tested, such as c0t3d0.
`partition=`<*0-7*>	Specifies the partition number to test as partition=6(/export/s6) if mounted on partition 6.
`rawsub=`*Enable/Disable*	Enables or disables the media subtest.
`rawrw=` *Readonly* *BackupWriteRead* *NoBackupWriteRead*	Specifies the Media subtest read/write mode: Read Only Write, read then backup Write, read without backup
`rawcover=`	Specifies media coverage from 0-100% of the partition, such as 70526f..
`rawiosize=`<*2KB,10KB 20KB,40KB,60KB*>	Specified the media size to transfer.
`fssub=`*Enable/Disable*	Enables or disables the file system subtest.
`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)}.
`fssize=`<*file system size*>	Indicates the file system subtest size in kilobytes or megabytes: K/k/KB/kb:kilobytes, M/m/MB/mb: megabytes
`fsiosize=`<*file system I/O transfer size*>	Indicates the size of the file system subtest I/O transfer in bytes or kilobytes: {512B/1024B/10KB/40KB/100KB}.

7

`disktest` *Error Messages*

`6000 Re-reading and recomparing block <number> on <name>.`

Probable_Cause(s):

a. Media error

b. Faulty cable/disk/controller

Recommended_Action(s):

a. If the problem persists, call your authorized Sun service provider.

`6002 Error on recomparing block <number> on <name>.`

Probable_Cause(s):

a. Media error

b. Faulty cable/disk/controller

Recommended_Action(s):

a. If the problem persists, call your authorized Sun service provider.

`6004 <name> failed on <name> '<name>', blk <number>: <error_message>.`

Probable_Cause(s):

a. Faulty cable/disk/controller

Recommended_Action(s):

a. If the problem persists, call your authorized Sun service provider.

`6006 Compare error on <name> <name>, block <number>, offset <number>.`

Probable_Cause(s):

a. Faulty cable/disk/controller

Recommended_Action(s):

a. If the problem persists, call your authorized Sun service provider.

```
6008 Compare error: Block <number> on <name> was written
with a repeating hex pattern of ‘<number>’.
```

Probable_Cause(s):

a. Media error

b. Faulty cable/disk/controller

Recommended_Action(s):

a. If the problem persists, call your authorized Sun service provider.

```
6010 TIME OUT!
```

Probable_Cause(s):

a. System too busy

b. Disk Faulty Cable/Disk

Recommended_Action(s):

a. Reduce system load

```
6012 Couldn’t close <name>.
```

Probable_Cause(s):

a. System error

```
6014 <name> read failed on disk, in-between blocks <number>
and <number>: <name>.
```

Probable_Cause(s):

a. Media error

b. Faulty cable/disk/controller

Recommended_Action(s):

a. If the problem persists, call your authorized Sun service provider.

```
6016 <name> read failed due to unexpected end of media:\
```

```
6018 <name> write failed on disk, in-between blocks
<number> and <number>: <name>.
```

Probable_Cause(s):

a. Media error

b. Faulty disk/controller

Recommended_Action(s):

a. If the problem persists, call your authorized Sun service provider.

```
6020 <name> write failed due to unexpected end of media:
```

```
6022 <name> compare failed on ‘<name>’, block <number>,
offset <number>
```

Probable_Cause(s):

a. Media error

b. Faulty disk/controller

Recommended_Action(s):

a. If the problem persists, call your authorized Sun service provider.

```
6024 Couldn’t close <name>.
```

Probable_Cause(s):

a. Faulty disk/controller

b. system error

Recommended_Action(s):

a. If the problem persists, call your authorized Sun service provider.

6026 failed automount <name> onto <name>:<error_message>

Probable_Cause(s):

a. Un-supported file system

b. System error

Recommended_Action(s):

a. Disable the file system subtest

b. Disable the file system subtest

c. If the problem persists, call your authorized Sun service provider.

6028 <error_message>

8000 File system subtest only allow to run on off-line mode

Probable_Cause(s):

a. Parameter error

Recommended_Action(s):

a. re-enter command line correctly

8002 malloc() failed

Probable_Cause(s):

a. System load too heavy

b. System error

8006 File operation error: Couldn't <name> file '<name>' on <name>: <error_message>.

Probable_Cause(s):

a. Lack of system resource

b. Permission error

c. System error

Recommended_Action(s):

a. Reduce the system load

b. Check directory/file permission

c. If the problem persists, call your authorized Sun service provider.

```
8008 Not enough free blocks on <name> (partition <number>)
```

Recommended_Action(s):

a. Try another partition or delete some files which no more needed

```
8010 No writable partition on <name>.
```

Recommended_Action(s):

a. Try another partition

```
8012 No file system on device: <name>.
```

Recommended_Action(s):

a. Use ‘newfs’ to create file system

b. Disable the file system subtest

```
8014 Couldn’t run fstest on any partition of device:
<name>(except 2)
```

Probable_Cause(s):

a. File system not supported

b. System error

Recommended_Action(s):

a. Disable the file system subtest

b. If the problem persists, call your authorized Sun service provider.

```
8016 Cannot open device: <name>.
```

Probable_Cause(s):

a. Parameter error

b. System error

Recommended_Action(s):

a. Enter the correct name of disk as dev=c0t3d0

b. If the problem persists, call your authorized Sun service provider.

```
8018 Couldn't mount /dev/dsk/<name>
```

Probable_Cause(s):

a. Parameter error

b. System error

Recommended_Action(s):

a. Enter the correct name of disk as dev=c0t3d0

b. If the problem persists, call your authorized Sun service provider.

```
8020 Couldn't remove temporary dir '<name>':
<error_message>.
```

Probable_Cause(s):

a. System error

Recommended_Action(s):

a. If the problem persists, call your authorized Sun service provider.

```
8022 fail get semaphore: <error_message>
```

Probable_Cause(s):

a. Lack of system resource

b. System error

Recommended_Action(s):

a. Please reduce system load

b. If the problem persists, call your authorized Sun service provider.

```
8024 fail lock mtab semaphore: <error_message>
```

Probable_Cause(s):

a. System error

Recommended_Action(s):

a. If the problem persists, call your authorized Sun service provider.

```
8026 fail unlock mtab semaphore: <error_message>
```

Probable_Cause(s):

a. System error

Recommended_Action(s):

a. If the problem persists, call your authorized Sun service provider.

```
8028 failed get_volmgr_name()

8030 Couldn't open <name>: <error_message>
```

Probable_Cause(s):

a. Cable loose or disconnected

b. Device off-line or missing

c. Device not configured

Recommended_Action(s):

a. Check cable

b. Check device on-line

c. Configure device

```
8032 Get file state <name> failed: <error_message>
```

Probable_Cause(s):

a. System error

Recommended_Action(s):

a. If the problem persists, call your authorized Sun service provider.

`8034 Couldn’t open <name>`

a. Probable_Cause(s):

Permission error

a. System error

Recommended_Action(s):

a. Check permission

b. If the problem persists, call your authorized Sun service provider.

`8036 failed putmntent(): <error_message>`

Probable_Cause(s):

a. System error

Recommended_Action(s):

a. If the problem persists, call your authorized Sun service provider.

`8038 fail rename file`

Probable_Cause(s):

a. Permission error

b. System error

Recommended_Action(s):

a. Check permission

b. If the problem persists, call your authorized Sun service provider.

`8040 Couldn’t get mount table entry for <name>.`

```
8042 Invalid device name in <name>
```

Probable_Cause(s):

a. Parameter error

Recommended_Action(s):

a. Enter the correct name of disk as dev=c0t3d0

```
8044 Message table overflow on device <name>
```

Probable_Cause(s):

a. Parameter error

b. System error

c. Program error

Recommended_Action(s):

a. Enter the correct name of disk as dev=c0t3d0

b. If the problem persists, call your authorized Sun service provider.

```
8046 Invalid controller id <name> for <name>
```

Probable_Cause(s):

a. Parameter error

Recommended_Action(s):

a. re-enter command line correctly

```
8048 <name>, disk DKIOCINFO ioctl: <name>
```

Probable_Cause(s):

a. Controller is not supported

b. System error

Recommended_Action(s):

a. Disable the file system subtest

b. If the problem persists, call your authorized Sun service provider.

```
8050 failed getting shared memory for monitor <number>:
<error_message>:
```

Probable_Cause(s):

a. Lack of system resource

b. System error

Recommended_Action(s):

a. Please reduce system load

b. If the problem persists, call your authorized Sun service provider.

```
8052 failed attaching monitor shared memory:<error_message>
```

Probable_Cause(s):

a. Lack of system resource

b. System error

Recommended_Action(s):

a. Please reduce system load

b. If the problem persists, call your authorized Sun service provider.

```
8053 failed create monitor shared memory <number>:
<error_message>
```

Probable_Cause(s):

a. Lack of system resource

b. System error

Recommended_Action(s):

a. Please reduce system load

b. If the problem persists, call your authorized Sun service provider.

```
8054 Shm remove error: <error_message>
```

Probable_Cause(s):

a. System error

Recommended_Action(s):

a. If the problem persists, call your authorized Sun service provider.

```
8056 New error messages in system log. Extracted
below:<error_message>
```

Probable_Cause(s):

a. Faulty cable/disk/controller

Recommended_Action(s):

a. See /var/adm/messages for more information

b. If the problem persists, consult your authorized Sun service provider.

```
8060 fails to open any partition of <name>
```

Probable_Cause(s):

a. File system not mounted

Recommended_Action(s):

a. Please mount the file system then run the test

b. Disable the file system subtest

c. If the problem persists, call your authorized Sun service provider.

```
8060 failed getting semaphore: <error_message>
```

Probable_Cause(s):

a. System error

Recommended_Action(s):

a. If the problem persists, call your authorized Sun service provider.

```
8062 failed getting shared memory: <error_message>
```

Probable_Cause(s):

a. System error

Recommended_Action(s):

a. If the problem persists, call your authorized Sun service provider.

`8064 failed attaching shared memory:<error_message>`

Probable_Cause(s):

a. System error

Recommended_Action(s):

a. If the problem persists, call your authorized Sun service provider.

`8068 failed unlocking semaphore: <error_message>`

Probable_Cause(s):

a. System error

Recommended_Action(s):

a. If the problem persists, call your authorized Sun service provider.

`8070 fail open floppy disk`

Probable_Cause(s):

a. No Floppy Disk in Drive

Recommended_Action(s):

a. Check the Floppy

`8072 failed FDIOGCHAR on floppy ioctl`

Probable_Cause(s):

a. System error

Recommended_Action(s):

a. If the problem persists, call your authorized Sun service provider.

```
8074 ioctl DKIOCGGEOM failed
```

Probable_Cause(s):

a. File system not supported

b. System error

Recommended_Action(s):

a. If the problem persists, call your authorized Sun service provider.

```
8074 ioctl DKIOCGAPART failed
```

Probable_Cause(s):

a. System error

Recommended_Action(s):

a. If the problem persists, call your authorized Sun service provider.

```
8076 fail initialize shared memory
```

Probable_Cause(s):

a. System error

Recommended_Action(s):

a. If the problem persists, call your authorized Sun service provider.

```
8078 Illegal parameters on Online Mode
```

Probable_Cause(s):

a. Parameter error

Recommended_Action(s):

a. Re-enter the correct option

8080 Illegal parameters on Conn. Mode

Probable_Cause(s):

a. Parameter error

Recommended_Action(s):

a. Re-enter the correct option

8082 failed get_volmgr_name()

Probable_Cause(s):

a. No Floppy Disk in Drive

Recommended_Action(s):

a. Check the Floppy

8084 Mounted file system on device!

Probable_Cause(s):

a. Not enough disk space

b. Permission error

c. System error

Recommended_Action(s):

a. Reduce the system load

b. Check directory/file permission

c. If the problem persists, call your authorized Sun service provider.

8086 Invalid controller id <number> from <name>

Probable_Cause(s):

a. Parameter error

Recommended_Action(s):

a. Enter the correct name of disk as dev=c0t3d0

8088 Couldn't get file system information on <name> errno=<number>

Probable_Cause(s):

a. System error

Recommended_Action(s):

a. If the problem persists, call your authorized Sun service provider.

8090 ioctl() failed on <name>: <name>.

Probable_Cause(s):

a. System error

Recommended_Action(s):

a. If the problem persists, call your authorized Sun service provider.

8092 Fail to get disk information

Probable_Cause(s):

a. No supported disk

b. Too heavy load system

Recommended_Action(s):

a. No UNIX message monitor feature in this disk(2)

b. Reduce system load

8094 Failed to create a message base for controller <number>

Probable_Cause(s):

a. No disk under this controller

Recommended_Action(s):

a. connect disk driver then run test again

```
8096 fail to make directory <name>
```

Probable_Cause(s):

a. Lack of system resource

b. Permission error

c. System error

Recommended_Action(s):

a. Reduce the system load

b. Check directory/file permission

c. If the problem persists, call your authorized Sun service provider.

```
8098 Failed to mount partition
```

Probable_Cause(s):

a. Does this disk only have partition 2

Recommended_Action(s):

a. Partition 2 must be manually mounted

```
8010 r/w is on /dev/r<name>
```

Probable_Cause(s):

a. System error

Recommended_Action(s):

a. If the problem persists, call your authorized Sun service provider.

7

ecpptest 8

*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.

Printer Test Hardware and Software Requirements

The ecpp(7) driver is installed with the OS 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.

8

ecpptest *Subtests*

Internal Test Fifo loopback

This verifies DMA and PIO accesses to the ecpp device. It utilizes 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 not available to an external customer.

Printer Test

This verifies the parallel port printer operation. It will output a half page of ascii character data. The output mode (ECP, Centronics, etc.) is determined by whatever 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.

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

ecpptest *Options*

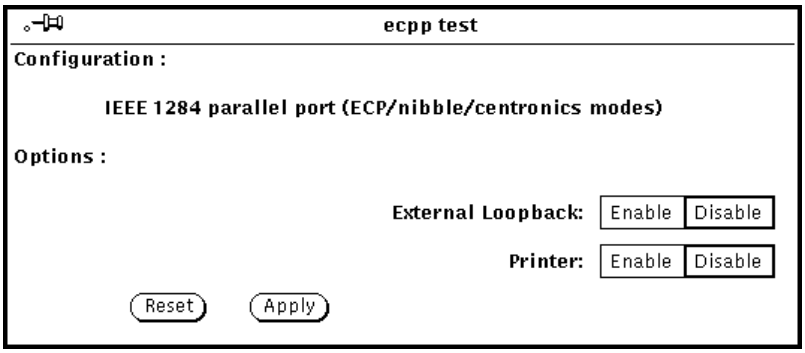


Figure 8-1 ecpptest Option Menu

- ***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.

ecpptest *Test Modes*

Connectivity Mode

In connectivity mode, a simple open and close of the ecpp(7) device is done. 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.

Online Mode

In online mode, the internal loopback test is run. If the device cannot be opened because it is busy, then it is assumed that the device is successfully connected to another process.

Offline Mode

In offline mode, the internal loopback test is run 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]`

Argument	Explanation
`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 /dec/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

`ecpptest` *Error Messages*

The `ecpptest` error messages are listed below with explanations or probable cause.

`8000 External loop test and printer test cannot be enabled at the same time.`

Probable_Cause(s):

a. Operator error

Recommended_Action(s):

a. Examine system message files (/var/adm/messages) for other information

`8001 <device_name>: Device does not exist or is otherwise inaccessible`

Probable_Cause(s):

a. **Correct** `ecpp` **driver not installed**

b. **Faulty** `ecpp` **device or motherboard hardware**

c. **System software problem**

Recommended_Action(s):

a. **Examine system message files (/var/adm/messages) for other information**

`8002 Must be super user (root) to execute`

Probable_Cause(s):

a. **Operator error**

Recommended_Action(s):

a. **Examine system message files (/var/adm/messages) for other information**

`8003 Failed to open <device_name> (<error_message>)`

Probable_Cause(s):

a. **Correct** `ecpp` **driver not installed**

b. **Faulty** `ecpp` **device or motherboard hardware**

c. **System software problem**

Recommended_Action(s):

a. **Examine system message files (/var/adm/messages) for other information**

`8004 Failed ioctl <name>, device = <device_name>, error = <error_message>`

Probable_Cause(s):

a. **Correct** `ecpp` **driver not installed**

b. **Faulty** `ecpp` **device or motherboard hardware**

c. **System software problem**

Recommended_Action(s):

a. **Examine system message files (/var/adm/messages) for other information**

```
8005 Device not in correct mode, device = <device_name>,
expected mode = <number>, observed mode = <number>
```

Probable_Cause(s):

a. **Correct** `ecpp` **driver not installed**

b. **Faulty** `ecpp` **device or motherboard hardware**

c. **System software problem**

Recommended_Action(s):

a. **Examine system message files (/var/adm/messages) for other information**

```
8006 <device_name>: Status line miscompare, expected =
<number>, observed = <number>, xor = <number>
```

Probable_Cause(s):

a. **Faulty or missing external loopback connector**

b. **Correct** `ecpp` **driver not installed**

c. **Faulty** `ecpp` **device or motherboard hardware**

d. **System software problem**

Recommended_Action(s):

a. **Examine system message files (/var/adm/messages) for other information**

```
8007 ECPP_nACK in wrong state, expected = <number>,
observed = <number>, xor = <number>
```

Probable_Cause(s):

a. **Faulty or missing external loopback connector**

b. **Correct** `ecpp` **driver not installed**

c. **Faulty** `ecpp` **device or motherboard hardware**

d. **System software problem**

Recommended_Action(s):

a. **Examine system message files (/var/adm/messages) for other information**

```
8008 <subtest_name>: Data miscompare, expected = <number>,
observed = <number>, xor = <number>
```

Probable_Cause(s):

a. **Faulty or missing external loopback connector**

b. **Correct** `ecpp` **driver not installed**

c. **Faulty** `ecpp` **device or motherboard hardware**

d. **System software problem**

Recommended_Action(s):

a. **Examine system message files (/var/adm/messages) for other information**

```
8009 Could not set correct port type, expected = <number>
(<device_name>), observed = <number>, device =
<device_name>
```

Probable_Cause(s):

a. **Correct** `ecpp` **driver not installed**

b. **System software problem**

c. **Faulty** `ecpp` **device or motherboard hardware**

Recommended_Action(s):

a. **Examine system message files (/var/adm/messages) for other information**

`8010 <subtest_name>: Failed to allocate <number> bytes of memory, error = <error_message>`

Probable_Cause(s):

a. **System resources temporarily not available, try again**

b. **Correct** `ecpp` **driver not installed**

c. **Faulty** `ecpp` **device or motherboard hardware**

d. **System software problem**

Recommended_Action(s):

a. **Examine system message files (/var/adm/messages) for other information**

`8011 <subtest_name>: Write() failed, device = <device_name>, error = <error_message>`

Probable_Cause(s):

a. **Correct** `ecpp` **driver not installed**

b. **Faulty** `ecpp` **device or motherboard hardware**

c. **System software problem**

Recommended_Action(s):

a. **Examine system message files (/var/adm/messages) for other information**

`8012 <subtest_name>: Write count is wrong, expected = <number>, observed = <number>`

Probable_Cause(s):

a. **Correct** `ecpp` **driver not installed**

b. **Faulty** `ecpp` **device or motherboard hardware**

c. **System software problem**

Recommended_Action(s):

a. **Examine system message files (/var/adm/messages) for other information**

```
8013 <subtest_name>: Data miscompare, expected = <number>,
observed = <number>, xor = <number>, offset = <number>
```

Probable_Cause(s):

a. **Correct** `ecpp` **driver not installed**

b. **Faulty** `ecpp` **device or motherboard hardware**

c. **System software problem**

Recommended_Action(s):

a. **Examine system message files (/var/adm/messages) for other information**

```
8014 <device_name>: Device busy
```

Probable_Cause(s):

a. **Another process is using the device**

b. **Someone is printing with the device**

c. **Faulty** `ecpp` **device or motherboard hardware**

d. **System software problem**

Recommended_Action(s):

a. **Make sure no other process is using the device**

b. **Make sure no one is printing to the device**

c. **Examine system message files (/var/adm/messages) for other information**

```
8015 Bad state for device <name>, expected mode = <number>,
observed mode = <number>.
```

8016 ecpp device busy: device <name>, expected mode = <number>, observed mode = <number>.

8017 Paper out: device <name>, expected mode = <number>, observed mode = <number>.

8018 Printer offline: device <name>, expected mode = <number>, observed mode = <number>.

8019 Printer error: device <name>, expected mode = <number>, observed mode = <number>.

8020 Printer unplugged: device <name>, expected mode = <number>, observed mode = <number>.

8021 Unrecognized printer error: device <name>, expected mode = <number>, observed mode = <number>.

fbtest 9

Frame Buffer Test (`fbtest`*)*

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

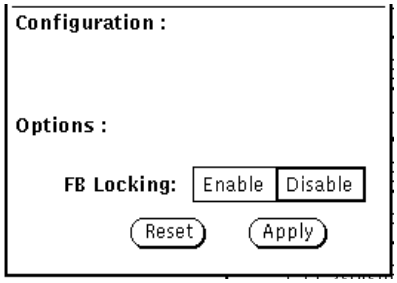


Figure 9-1 `fbtest` Option Menu

9

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

`fbtest` *Command Line Syntax*

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

Argument	Explanation
`dev`=*device_name*	Specifies which frame buffer to be tested
`lock=`*E(nable)/D(isable)*	Enables / disables the window system locking option. See the section about Testing Multiple Frame Buffers in *SunVTS 2.0 User's Guide* for details. Frame buffer locking is enabled by default on the window server running the OpenWindows software.

`fbtest` *Test Modes*

Connectivity Mode

Not applicable for this test.

Online Mode

Not applicable for this test.

Offline Mode

Due to the nature of graphic tests, reading from or writing to the frame buffer during graphic tests disturbs user operation. This test is only available in Offline Mode.

`fbtest` *Error Messages*

The `fbtest` error messages are listed below with explanations or probable causes:

`6000 Data compare error`

Probable_cause(s):

a. FB lock disabled

b. Bad Frame Buffer

Recommended_Action(s):

a. Enable FB lock

b. If the problem persists, call your authorized Sun service provider.

`6001 Video memory error at <address>, expected: <number>, observed: <number>`

Probable_cause(s):

a. FB lock disabled

b. Bad Frame Buffer

Recommended_Action(s):

a. Enable FB lock

b. If the problem persists, call your authorized Sun service provider.

`6002 Unable to lock frame buffer.`

Probable_cause(s):

a. FB lock was enabled while Window System was not running

`8000 can’t acquire console semaphore`

Probable_cause(s):

a. System load too heavy

b. System error

Recommended_Action(s):

a. Reduce the system load

b. If the problem persists, contact your Sun service provider

```
8001 Parameter error - Improper test parameters supplied.
```

Probable_cause(s):

a. fbtest can only run in offline mode

```
8002 Device name missing!
```

Probable_cause(s):

a. Parameter error.

Recommended Action(s):

a. Must provide device name

```
8003 Create_Raster() failed for <name>
```

Probable_cause(s):

a. Parameter error

b. No such device

```
8004 Create_Raster() failed for type RAS_CHILD
```

Probable Causes(s):

a. Operator error

b. Software error

Recommended Action(s):

a. Check test options, mode and retry if appropriate

b. Consult Support Representative

```
8005 Malloc failed.
```

Probable_cause(s):

a. System load too heavy

b. System error

Recommended_Action(s):

a. Reduce the system load

b. If the problem persists, call your authorized Sun service provider.

9

ffbtest 10

*Fast Frame Buffer Test (*ffbtest*)*

The ffbtest can detect and adapt to the video modes of single- and double-buffer versions of the fast frame buffer (FFB). 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 ffbtest using Control-C. Turn off all other keyboard inputs 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 FFB accelerator port while running ffbtest. This combination causes 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.

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

Note – `ffbtest` requires approximately 2 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.

`ffbtest` *Options*

By default, all tests are enabled.

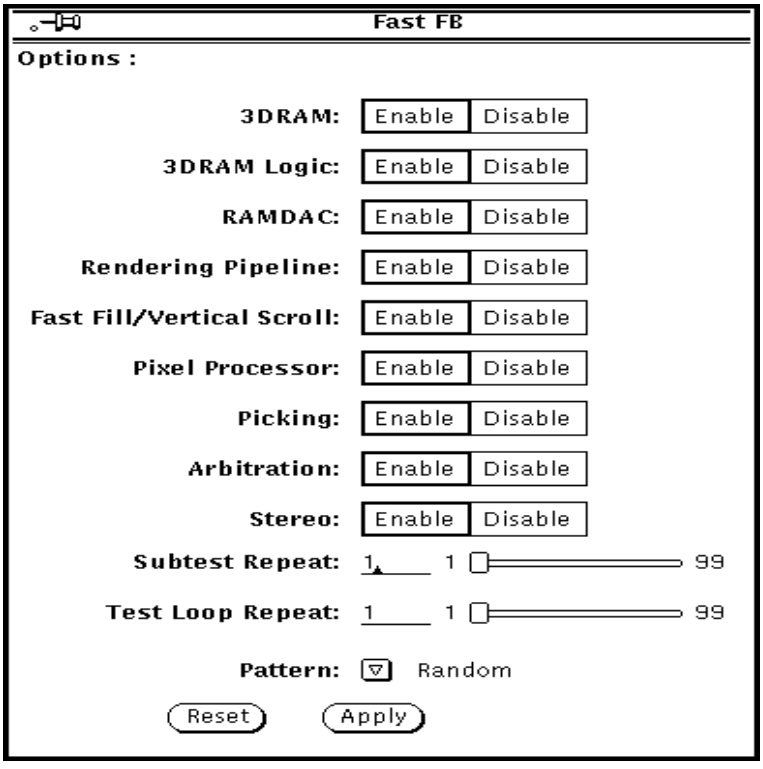


Figure 10-1 `ffbtest` Option Menu

3DRAM Test Description

3DRAM 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. The data used can be either random or specified by the user 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 & C (double buffer only)
- SFB64 - Buffers B & C (double buffer only)

3DRAM Logic Test Description

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)

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 to be tested.

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 which 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.

RAMDAC Test Description

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)

A total of 11 different types of windows appear on the screen for the single-buffered configuration; 22 for double-buffered. A cursor also appears on the screen.

Errors in this test are attributed to the RAMDAC.

Rendering Pipeline Test Description

Rendering Pipeline uses the rendering pipeline tests developed for the FFB stand-alone diagnostics. Each 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 FBC

Fast Fill/Vertical Scroll Test Description

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 FBC.

Pixel Process Test Description

Pixel Processor, a subtest, exercises the 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.

Picking Test Description

Picking 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 Description

Arbitration, a subtest, continuously renders an object into the accelerator port while doing reads and writes through the direct port. For single buffered configuration, a picture is rendered into the RGB planes while another process does DFB reads and writes in the X plane. For double buffered, 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 FBC.

Stereo Test Description

Stereo 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 @ 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 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 a display problem such as the one described above, disable the `ffbtest` stereo test when other SunVTS tests are enabled.

`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 mode.

Connectivity Mode

Not applicable for this test.

Online Mode

Not applicable for this test.

Offline Mode

The `ffbtest` verifies both the single (SFB) and double buffer (DBZ) fast frame buffer boards.

Stress Mode

The Stress Mode exercises the device being tested 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. Then the primitives are checked to ensure that they were rendered the same as on a known good system. The test is repeated ten times, with each random picture overlaying the previous one.

`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*

Argument	Explanation
dev=*device_name*	*device_name* is the full path name of the device being tested; the default is `/dev/fbs/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 means 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; default is 1.
B=*#_of_test_loops*	Specifies the number of times to repeat a test loop before passing; 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**.

10

`ffbtest` *Error Messages*

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. The error messages are listed below, along with explanations and probable cause.

Error messages 1-7 all occur during the screen locking procedure where ffbtest is interacting with the X window system. The test only supports FFB devices on display 0 - most machines only have display with multiple screens, but it is possible to have multiple displays. Make sure you are using OPEN LOOK windows from SunSoft - other window systems are not supported and may not work.

1. Can't acquire console semaphore

2. Cannot gain control of mouse: *<reason>*

3. Cannot gain control of keyboard: *<reason>*

4. DGA error grabbing window on screen *<value>*.

5. DGA error getting window from token on screen *<value>*.

6. No FFB device found on display 0.

7. There is an inconsistency with the version of the X Window System.

8. Can't access XWindows - may need to run xhost(1).

   The ffbtest process must have write access to the X Windows display on the FFB. To grant access, run 'xhost +'.

9. <*test name*> error: 3DRAM Memory error
   <*plane group*> Plane at (<*x*>,<*y*>) Expected = <value>, actual = <*value*>, xor = <*value*>.

   The value written to the frame buffer was not the same as the value read back. This usually indicates a faulty 3DRAM chip. The u-number (FFB board position) of the failing chip is indicated in the error message. This message may be reported by the 3DRAM test or the arbitration test.

10. <test name>: Down-revved 3DRAM single bit failure detected <plane group> Plane at (<*x*>,<*y*>) Expected = <*value*>, actual = <*value*>, xor = <*value*>.

    Early production 3DRAM chips exhibited single-bit read back errors on this test. This warning message indicates that your FFB has one of these chips, and the error was detected. The u-number (FFB board position) of the failing chip is indicated in the error message. This message may be reported by the 3DRAM test or the arbitration test.

11. <test name> error.
    Miscompares, starting at %s, are colored chartreuse.

    The image generated by the test does not match the image generated on a known good FFB. The pixels which are colored chartreuse are where the images don't agree. Because of the checksum algorithm used for image comparison, some of the colored pixels may actually have been correct, but incorrect pixels obscured the comparison.

12. Installed FFB has downrev RAMDAC - skipping RAMDAC test

    Early versions of the FFB RAMDAC chip had faulty signature analysis and therefore could not be used for the RAMDAC test. These chips are not necessarily bad, but they cannot be tested with ffbtest.

13. Unsupported screen resolution - skipping RAMDAC test

    Almost all screen resolutions are supported by the RAMDAC test, but a few are not.

14. Miscompare found in <Red Green Blue> channel of CLUT entry *<value>*.
    Expected = *<value>*, Observed = *<value>*, XOR = *<value>*.

15. Miscompare found in WID LUT entry *<value>*.
    Expected = *<value>*, Observed = *<value>*, XOR = *<value>*.

    The RAMDAC test writes values to each entry of the lookup tables and then reads back the values. These error messages indicate that the readback value was different from the written value. This could indicate a faulty RAMDAC.

16. RAMDAC test failed. The on-screen image is incorrect.

    The RAMDAC test puts up squares of data using all of its different output modes. It then gets a “signature” from the RAMDAC, which is a checksum of the pixels actually being written to the screen. This message indicates that the checksum is different from the checksum generated on a known good FFB. This could indicate a faulty RAMDAC.

    Error messages 17-19 indicate a failure in the picking test. On the FFB, picking is done by the 3DRAM chips, so the u-number (FFB board position) of the failing chip is indicated in the error message.

17. Picking test error.

    Got a pick hit with picking disabled.

18. Picking test error.

    Got a pick hit outside the picking area.

19. Picking test error.

Should have detected a pick hit.

Error messages 20-22 indicate incorrectly installed or corrupted software installation. Try reinstalling SunVTS.

20. Stereo image incorrect.

The Stereo test uses the RAMDAC signature analysis feature to verify that the pixels going to the screen are correct. This message indicates a failure of the test, which probably indicates a faulty RAMDAC.

21. Stereo not supported on current monitor type: *<value>*.

Only certain kinds on monitors support Stereo output mode. Only the P4 monitors will work with the ffbtest Stereo test. File Not Found
Could not open file <file name>.

22. Could not open file *<filename>*: *<reason>*.

Keyword not Found, line *<value>*
Suspect data file corruption - reinstall software.

23. Out of memory.

A call to mailloc(3c) or new( ) failed. Increase swap space.

24. /usr/bin/tar failed.

Check for existence and/or permissions on <file name>. Also check for available space in /tmp.

Error messages 25-35 indicate that there is a permissions problem. You must be root to run ffbtest. These messages may also indicate some other kind of corruption. Try rebooting, and if that fails, reinstalling.

25. /usr/bin/uncompress failed.

Check for existence and/or permissions on <file name>.

26. fork1(2) failed: *<reason>*

27. execl(2) failed: *<reason>*

28. sysinfo(2) failed getting hostname. Must be superuser.

29. mmap(2) failed: *<reason>*.

30. munmap(2) failed: *<reason>*.

31. close(2) failed: *<reason>*.

    Unable to open <d*evice name*>. Check device for existence and/or permission.

32. wait(2) failed: *<reason>*

33. signal(2) failed: *<reason>*

34. thr_kill(2) failed

35. thr_join(2) failed: *<reason>*

    Error messages 36-51 indicate either that the FFB was installed incorrectly or that you do not have permission to use it. Check that the FFB packages were installed correctly and that you have the latest version of the driver patches. Also make sure you are running ffbtest as root.

36. ioctl(2) failed trying to read CLUT.

37. ioctl(2) failed trying to turn on FFB video.

38. ioctl(2) failed trying to post CLUT.

39. ioctl(2) failed trying to read WID LUT.

40. ioctl(2) failed trying to post WID LUT.

    Suspect downrev ffb driver or very old electron board.

41. ioctl(2) failed trying to get RAMDAC signature analysis.

42. ioctl(2) failed trying to get current video mode.

43. ioctl(2) failed trying to set current video mode.

44. ioctl(2) failed trying to get FFB system info.

45. ioctl(2) failed trying to go into Diagnostic mode.

46. ioctl(2) failed trying to read Transparent Overlay registers.

47. ioctl(2) failed trying to post Transparent Overlay registers.

48. ioctl(2) failed trying to post cursor.

49. ioctl(2) failed trying to read cursor.

50. stat(2) failed trying to get info about Device Under Test.

51. stat(2) failed trying to get info about screen 0:*<value>*.

10

fputest 11

Floating Point Unit Test (fputest)

This test checks the floating point unit on machines with the SPARC® architecture. `fputest` performs the following 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
- Cora test
- Kcsqrt test
- Kcdiv test
- Clorenz test
- Cvector test

`fputest` *Options*

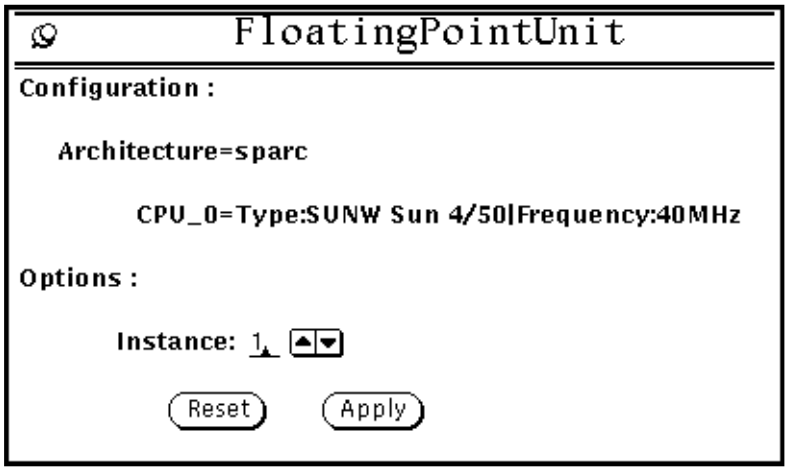


Figure 11-1 `fputest` Option Menu

`fputest` *Test Modes*

Connectivity Mode

In this mode, `fputest` includes all the Instruction Tests.

Online Mode

In this mode, `fputest` includes Linpack test and all the Instruction Tests.

Offline Mode

Offline mode includes all the Instruction Tests and all the Benchmark Tests.

Stress Mode

When stress mode is selected, several fpu benchmark tests are run.

`fputest` *Command Line Syntax*

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

`fputest` *Error Messages*

Error messages that describe the math operation or instruction being tested and the expected and observed values are listed below:

```
6012 Precision worse than 5 decimal figures.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6013 Radix is too big: roundoff problems.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6014 Radix is not as good as 2 or 10.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6015 (1-U1)-1/2 < 1/2 is FALSE, prog. fails?
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6016 X=1 but X-1/2-1/2 != 0.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6017 Subtraction is not normalized X=Y,X+Z != Y+Z!
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6018 Multiplication lacks a Guard Digit, so 1*X != X.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6019 Multiplication gets too many final digits wrong.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6020 Division lacks a Guard Digit.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6021 Division lacks a Guard Digit, so X/1 != X.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6022 Computed value of 1/1.000..1 >= 1.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

6023 Multiplication and/or Division gets too many last digits wrong.

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

6024 Subtraction lacks Guard Digit, so cancellation is obscured.

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

6025 X * (1/X) differs from 1.

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

6026 Radix * (1 / Radix) differs from 1.

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

6027 Incomplete carry-propagation in Addition.

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6028 Square root of 0.0, -0.0 or 1.0 wrong.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6029 sqrt gets too many last digits wrong.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6030 Square root is neither chopped nor correctly rounded.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6031 PseudoZero prints out as: <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6032 -PseudoZero prints out as: <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6033 Underflow prints out as: <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6034 Accuracy deteriorates as numbers approach a threshold
= <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6035 Underflow confuses Comparison.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

6036 X = <value> is not equal to Z = <value> yet X - Z yields <value>.

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

6037 X = <value> is not equal to Z = <value> yet X - Z yields <value>.

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

6038 Calculation yields: <value> :This is not between 0 and underflow threshold = <value>.

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

6039 Calculation yields: <value> :This is not between 0 and underflow threshold = <value>.

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

6040 Calculated <value> for (1 + (<value>) ^ (<value>); differs from correct value by <value>.

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

6041 Calculated <value> for (1 + (<value>) ^ (<value>); differs from correct value by <value>.

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

6042 Overflow(`Z = -Y') test on Y = <value> finds that -(-Y) differs from Y.

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

6043 Overflow(`Z = -Y') test on Y = <value> finds that -(-Y) differs from Y.

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

6044 Overflow past <value> shrinks to <value>.

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

6045 Overflow past <value> shrinks to <value>.

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

6046 Comparisons involving +-<value>, +-<value>and +-<value> are confused by Overflow.

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

6047 Comparisons involving +-<value>, +-<value>and +-<value> are confused by Overflow.

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

6048 A total of <value> floating point exceptions were registered.

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6049 (<value>) ^ (<value>) yielded <value>; which compared
unequal to correct <value>; they differ by <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6050 (<value>) ^ (<value>) yielded <value>; which compared
unequal to correct <value>; they differ by <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6051 What prints as Z = <value> compares different from Z
* 1 = <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6052 What prints as Z = <value> compares different from 1
* Z == <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6053 What prints as Z = <value> compares different from Z
/ 1 = <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6054 What prints as Z = <value> compares different from Z
* 1 = <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6055 What prints as Z = <value> compares different from 1
* Z == <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6056 What prints as Z = <value> compares different from Z
/ 1 = <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6057 Multiplication does not commute:Comparison alleges
that 1 * Z = <value> differs from Z * 1 = <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6058 Multiplication does not commute:Comparison alleges
that 1 * Z = <value> differs from Z * 1 = <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6059 <subtest_name> test appears to be inconsistent...
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6060  Comparison alleges that -0.0 is Non-zero!
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6061  Disagreements among the values X1, Y1, Z1
respectively  <value>  <value>  <value>
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6062  Disagreements among the values X1, Y1, Z1
respectively  <value>,  <value>,  <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6063  Comparison alleges  (1-U1) < 1  although subtraction
yields  (1-U1) - 1 = 0.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6064  Multiplication appears to chop.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6065  Multiplication is neither chopped nor correctly
rounded.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6066  Division appears to chop.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6067  Division is neither chopped nor correctly rounded.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6068  Addition/Subtraction appears to be chopped.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6069  Addition/Subtraction neither rounds nor chops.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6070  (X - Y) + (Y - X) is non zero.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6071  Sticky bit used incorrectly or not at all.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6072  X * Y == Y * X trial fails.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6073  Sqrt(X) is non-monotonic for X near <value>
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6074  Sqrt(X) is non-monotonic for X near <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6075  Anomalous arithmetic with Integer < Radix^Precision =
<value> fails test whether sqrt rounds or chops.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6076  Anomalous arithmetic with Integer < Radix^Precision =
<value>  fails test whether sqrt rounds or chops.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6077  Square root appears to be chopped.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

6078 Comparison alleges that Z = <value> is too far from sqrt(Z) ^ 2 (<value>).

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

6079 Comparison alleges that Z = <value> is too far from sqrt(Z) ^ 2 (<value>).

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

6080 Unbalanced range; UfThold * V = <value> is too far from 1.

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

6081 Unbalanced range; UfThold * V = <value> is too far from 1.

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6082  Sqrt( <value>) - <value>  = <value> instead of
correct value 0.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6083  Sqrt( <value>) - <value>  = <value> instead of
correct value 0.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6084  Similar discrepancies have occurred <value> times.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6085  X / X  traps when X = <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6086  X / X  traps when X = <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6087  X / X differs from 1 when X = <value> instead, X / X
- 1/2 - 1/2 = <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6088  X / X differs from 1 when X = <value> instead, X / X
- 1/2 - 1/2 = <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6089  Computed:<character>   Expected:<character>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6090  Failed Vector test.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6091  Error : Invalid operator (<character>).
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6092  Error : Invalid version (<character>).
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6093  Error : Invalid precision (<character>).
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6094  Error : Illegal input (<character>).
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6095  Failed Vector <subtest_name> sub-
test[<text>]\n                  Operand 1 <text>  <value>
<value>\n                    Operand 2 <text>  <value>
<value>\n                 <character><character><character>
Correct   <text>  <value>  <value>
(<text>)\n                   Computed  <text>  <value>
<value>  (<text>).
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6096  Failed Vector <subtest_name> sub-
test[<text>]\n                  Operand 1 <text>  <value>
<value> <value>\n                Operand 2 <text>  <value>
<value> <value>\n
<character><character><character> Correct  <text>  <value>
<value> <value>  (<text>)\n                    Computed
<text>  <value>  <value> <value>  (<text>).
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6097  Failed Vector <subtest_name> sub-
test[<text>]\n                   Operand 1 <text>  <value>
<value> <value> <value>\n                Operand 2 <text>
<value> <value> <value> <value>\n
<character><character><character> Correct   <text>  <value>
<value> <value> <value>  (<text>)\n              Computed
<text>  <value>  <value> <value> <value>  (<text>).
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6098  Failed Vector <subtest_name> sub-
test[<text>]\n                    Operand 1 <text>  <value>
<value> <value>\n                  Operand 2 <text>  <value>
<value> <value>\n
<character><character><character> Correct   <text>  <value>
<value> <value>  (<text>)\n                    Computed
<text>  <value>  <value> <value>  (<text>).
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6099  <character><character><character> <value> (<text>)
<character> <value> (<text>) <computed <character>>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6100  <character><character><character> Op1 (<text>)
<character> Op2 (<text>) <computed <character>>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6101  Flags : Correct (<text>) Computed (<text>).
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6102  FPU initialization failure.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6103  Multiplication failed, result was (<value>), expected
(<value>).
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6104  Failed single precision FPA math test.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6105  Failed double precision FPA math test.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6106  Failed linpack test.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6107  Failed cparanoia test.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6108  Failed cora test.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

6109 Failed clorenz test.

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6110  Failed kcsqrt test.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6111  Failed kcdiv test.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6112  Error:  a + b\nExpected: 2.2221000 Actual: <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6113  Error   a - b\nExpected: 0.2469000 Actual: <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6114  Error   a * b\nExpected: 1.2191922 Actual: <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6115  Error    a / b\nExpected: 1.2500000 Actual: <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6116  Error:  a + (a + b)\nExpected: 1.4814000
Actual: <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6117  Error:  a - (a + b)\nExpected: -0.9876000
Actual: <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6118  Error:  a + (a * b)\nExpected: 2.4536924
Actual: <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6119  Error:  a - (a * b)\nExpected: 0.0153078
Actual: <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6120  a + (a / b)\nExpected: 2.4845002 Actual: <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6121  Error:  a - (a / b)\nExpected: -0.0155000
Actual: <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6122  Error:  a * (a + b)\nExpected: 2.7431825
Actual: <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6123  Error:  a * ( a - b)\nExpected: 0.3047980
Actual: <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6124  Error:  a / ( a - b)\nExpected: 0.5555550
Actual: <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6125  Error:  a / ( a - b)\nExpected: 5.0000000
Actual: <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6126  Error:  a * ( a / b)\nExpected: 1.5431250
Actual: <value>).
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6127  Error:  a / ( a * b)\nExpected: 1.0125557
Actual: <value>).
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6128  Error:  a + b\nExpected: 2.222100000000000
Actual: <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6129  Error:  a - b\nExpected: 0.246899999999999
Actual: <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6130  Error:  a * b\nExpected: 1.219192199999999
Actual: <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6131  Error:  a / b\nExpected: 1.249999999999999
Actual: <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6132  Error:  a + (a - b)\nExpected:
1.481399999999999       Actual: <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6133  Error:  a - (a + b)\nExpected: -
0.987600000000000      Actual: <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6134  Error:  a + (a * b)\nExpected:
2.453692200000000       Actual: <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6135  Error:  a - (a * b)\nExpected:
0.015307800000000       Actual: <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6136  Error:  a + (a / b)\nExpected:
2.484500000000000       Actual: <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6137  Error:  a - (a / b)\nExpected: -
0.015499999999999      Actual: <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6138  Error:  a * (a + b)\nExpected:
2.743182449999999       Actual: <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6139  Error:  a * (a - b)\nExpected:
0.304798049999999       Actual: <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6140  Error:  a / (a + b)\nExpected:
0.555555555555555       Actual: <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6141  Error:  a / (a - b)\nExpected:
5.000000000000002       Actual: <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6142  Error:  a * (a / b)\nExpected:
1.543124999999999       Actual: <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6143  Error:  a / (a * b)\nExpected:
1.0125555690562980      Actual: <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6144  Error: sin(-2pi)\nExpected: -
0.000000000820413          Actual: <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6145  Error: sin(-3pi/2)\nExpected:
1.0000000000000000       Actual: <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6146  Error: sin(-3pi/2)\nExpected:
1.0000000000000000       Actual: <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6147  Error: sin(-pi)\nExpected:
0.000000000410206            Actual: <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6148  Error: sin(-pi/2)\nExpected: -
1.0000000000000000       Actual: <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6149  Error: sin(0)\nExpected:
0.0000000000000000             Actual: <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6150  Error: sin(pi/2)\nExpected:
1.0000000000000000         Actual: <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6151  Error: sin(pi)\nExpected: -
0.000000000410206           Actual: <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6152  Error: sin(3pi/2)\nExpected: -
1.0000000000000000       Actual: <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6153  Error: sin(2pi)\nExpected:
0.000000000820143           Actual: <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6154  Error: cos(-2pi)\nExpected:
1.0000000000000000         Actual: <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6155  Error: cos(-3pi/2)\nExpected: 0.000000000615310
Actual: <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6156  Error: cos(-pi)\nExpected: -1.0000000000000000
Actual: <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6157  Error: cos(-pi/2)\nExpected: -0.000000000205103
Actual: <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6158  Error: cos(0)\nExpected: 1.0000000000000000
Actual: <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6159  Error: cos(pi/2)\nExpected: -0.000000000205103
Actual: <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6160  Error: cos(pi)\nExpected: -1.0000000000000000 Actual:
<value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6161  Error: cos(3pi/2)\nExpected: 0.000000000615310
Actual: <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6162  Error: cos(pi/2)\nExpected: 1.0000000000000000
Actual: <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6163  Error: sin(pi/4)\nExpected: 0.707106781259062 Actual:
<value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6164  Error: sin(3pi/4)\nExpected: 0.707106780969002
Actual: <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6165  Error: sin(5pi/4)\nExpected: -0.707106781549122
Actual: <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6166  Error: sin(7pi/4)\nExpected: -0.707106780678942
Actual: <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6167  Error: cos(pi/4)\n Expected: 0.707106781114032
Actual: <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6168  Error: cos(3pi/4)\n Expected: -0.707106781404092
Actual: <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6169  Error: cos(5pi/4)\n Expected: -0.707106780823972
Actual: <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6170  Error: cos(7pi/4)\n Expected: 0.707106781694152
Actual: <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6171  Error: exp(0)\n Expected: 1.0000000000000000 Actual:
<value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6172  Error: exp(1)\n Expected: 2.718281828459045 Actual:
<value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6173  Error: exp(2)\n Expected: 7.389056098930650 Actual:
<value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6174  Error: exp(5)\n Expected: 148.413159102576600 Actual:
<value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6175  Error: exp(10)\n Expected: 22026.465794806718000
Actual: <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6176  Error: exp(-1)\n Expected: 0.367879441171442 Actual:
<value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6177  Error: exp(-2)\nExpected: 0.135335283236612 Actual:
<value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6178  Error: exp(-5)\nExpected: 0.006737946999085 Actual:
<value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6179  Error: exp(-10)\nExpected: 0.000045399929762 Actual:
<value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6180  Error: exp(log(1)\nExpected: 1.0000000000000000
Actual: <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6181  Error: exp(log(10)\nExpected 10.000000000000002
Actual: <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6182  Error: log(1)\nExpected: 0.0000000000000000 Actual:
<value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6183  Error: log(2)\nExpected: 0.693147180559945 Actual:
<value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6184  Error: log(10)\nExpected: 2.302585092994045 Actual:
<value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6185  Error: log(100)\nExpected: 4.605170185988091 Actual:
<value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6186  Error: log(exp(0))\nExpected: 0.0000000000000000
Actual: <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6187  Error: log(exp(1))\nExpected: 1.0000000000000000
Actual: <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6188  Error: log(exp(10))\nExpected: 10.0000000000000000
Actual: <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6189  Error: tan(-2pi)\nExpected: -0.000000000820414
Actual: <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6190  Error: tan(-7pi/4)\nExpected: 0.999999998564275
Actual: <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6191  Error: tan(-5pi/4)\nExpected: -1.000000001025517
Actual: <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6192  Error: tan(-pi\nExpected: 0.000000000410207  Actual:
<value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6193  Error: tan(-3pi/4)\nExpected: 0.999999999384690
Actual: <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6194  Error: tan(-pi/4)\nExpected: -1.000000000205103
Actual: <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6195  Error: tan(0.0)\nExpected: 0.000000000000000  Actual:
<value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6196  Error: tan(pi / 4)\nExpected: 1.000000000205103
Actual: <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6197  Error: tan(3pi/4)\nExpected: -0.999999999384690
Actual: <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6198  Error: tan(pi)\nExpected: 0.000000000410207  Actual:
<value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6199  Error: tan(5pi/4)\nExpected: 1.000000001025517
Actual: <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6200  Error: tan(7pi/4)\nExpected: -0.999999998564275
Actual: <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6201  Error: tan(2pi)\nExpected: 0.000000000820414  Actual:
<value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6202  Error: sqrt(0)\nExpected: 0.000000000000000  Actual:
<value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6203  Error: sqrt(1)\nExpected: 1.000000000000000  Actual:
<value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6204  Error: sqrt(4)\nExpected: 2.000000000000000  Actual:
<value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6205  Error: sqrt(9)\nExpected: 3.000000000000000  Actual:
<value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6206  Error: sqrt(16)\nExpected: 4.000000000000000  Actual:
<value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6207  Error: sqrt(25)\nExpected: 5.000000000000000  Actual:
<value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6208  Error: sqrt(36)\nExpected: 6.000000000000000  Actual:
<value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6209  Error: sqrt(49)\nExpected: 7.000000000000000  Actual:
<value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6210  Error: sqrt(64)\nExpected: 8.000000000000000  Actual:
<value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

`6211 Error: sqrt(81)\nExpected: 9.000000000000000 Actual: <value>.`

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

`6212 Error: sqrt(100)\nExpected: 10.000000000000000 Actual: <value>.`

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

`6213 log-test failed: int = <value>, expected / observed = <value> / <value>.`

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

`6214 kvm_read(mem,nl[0].n_value,...) failed`

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6215  kvm_open() failed:<error_message>
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6216  kvm_nlist() failed:<error_message>
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6217  Add SP failed: expected / read = <value> / <value>
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6218  Multiply SP failed:expected / read = <value> /
<value>
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6219  Add DP failed: msw : expected / read = <value> /
<value>\n                  lsw : expected / read = <value>
/ <value>
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6220  Multiply DP failed: msw : expected / read = <value>
/ <value>\n                 lsw : expected / read = <value>
/ <value>
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6221  Single Precision: add, expected / observed =
0x41200000 / 0x<value>
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6222  Single Precision: Multiply, expected / observed =
0x43470000 / 0x<value>
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6223  Double Precision: Add, MSW : expcected / observed =
0x40240000 / 0x<value>\n                              LSW :
expected / observed = 0x0 / 0x<value>
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6224  Double Precision: Multiply, MSW : expected / observed
= 0x4034000 / 0x<value>\n                                 LSW
: expected / observed = 0x0 / 0x<value>
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6225  Error: expected / observed = <value> / 0x<value>
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6226  Error: expected / observed = <value> / 0x<value>
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6227  fitos failed: int = <value>, expected / observed =
<value> / <value>
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6228  fitod failed: int = <value>, expected / observed =
<value> / <value>
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6229  fstoi failed: int = <value>, expected / observed =
<value> / <value>
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6230  fdtoi failed: int = <value>, expected / observed =
<value> / <value>
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6231  fstod failed: int = <value>, expected / observed =
<value> / <value>
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6232  fdtos failed: int = <value>, expected / observed =
<value> / <value>
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6233  fmovs failed : written <value> to f0, read from f31
= <value>
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6234  fnegs failed(from pos to neg): int = <value>,
expected / observed =  <value> / <value>
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6235  fnegs failed (from neg. to pos): int = <value>,
expected / observed = <value> / <value>
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6236  fabs failed: int = <value>, expected / observed =
<value> / <value>
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6237  fadds failed: int = <value>, f0 = <value>, f2 =
<value>, f0+f2 = f4 = <value>
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6238  faddd failed: int = <value>, f0 = <value>, f2 =
<value>, f0+f2 = f4 =  <value>
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6239  fsubs failed:int = <value>, f0 = <value>, f2 =
<value>, f0-f2 = f4 = <value>
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6240  fsubd failed: int = <value>, f0 = <value>, f2 =
<value>, f0-f2 = f4 = <value>
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6241  fsqrt(single) failed: written / read = <value> /
<value>
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6242  fsqrt(double) failed: written / read = <value> /
<value>
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6243  fdivs failed: int = <value>, f0 = <value>, f2 =
<value>, f0 / f2 = f4 = <value>
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6244  fdivd failed: int = <value>, f0 = <value>, f2 =
<value>, f0 / f2 = f4 = <value>
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6245  fmuls failed: int = <value>, f0 = <value>, f2 =
<value>, f0 / f2 = f4 = <value>
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6246  fmuld failed: int = <value>, f0 = <value>, f2 =
<value>, f0 / f2 = f4 = <value>
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6247  fcmps failed: f0 = <value>, f2 = <value> : expected
/ observed = 0 / <value>
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6248  fcmps failed: f0 = <value>, f2 = <value> : expected
/ observed = 1 /<value>
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6249  fcmps failed: f0 = <value>, f2 = <value> : expected
/ observed = 2 /<value>
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6250  fcmps failed: f0 = <value>, f2 = NaN : expected /
observed = 3 /<value>
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6251  fcmpd failed: f0 = <value>, f2 = <value> : expected
/ observed = 0 / <value>
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6252  fcmpd failed: f0 = <value>, f2 = <value> : expected
/ observed = 1 /<value>
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6253  fcmpd failed: f0 = <value>, f2 = <value> : expected
/ observed = 2 /<value>
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6254  fcmpd failed: f0 = <value>, f2 = NaN : expected /
observed = 3 /<value>
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6255  FBU failed. result = <value>
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6256  FBG failed: f0 = <value>, f2 = <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6257  FBUG (unordered) failed.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6258  FBUG (greater) failed: f0 = <value>, f2 = <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6259  FBL failed: f0 = <value>, f2 = <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6260  FBUL (unordered) failed.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6261  FBUL (Less) failed: f0 = <value>, f2 = <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6262  FBLG (Less) failed: f0 = <value>, f2 = <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6263  FBLG (Greater) failed: f0 = <value>, f2 = <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6264  FBNE failed: f0 = <value>, f2 = <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6265  FBE failed : f0 = <value>, f2 = <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6266  FBUE (unordered) failed.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6267  FBUE (equal) failed : f0 = <value>, f2 = <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6268  FBGE (equal) failed : f0 = <value>, f2 = <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6269  FBGE (greater) failed: f0 = <value>, f2 = <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6270  FBUGE (unordered) failed.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6271  FBUGE (equal) failed : f0 = <value>, f2 = <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6272  FBUGE (greater) failed: f0 = <value>, f2 = <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6273  FBLE (Less) failed: f0 = <value>, f2 = <value>.
```

Probable_Cause(s):

b. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6274  FBLE (equal) failed : f0 = <value>, f2 = <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6275  FBULE (unordered) failed.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6276  FBULE (Less) failed: f0 = <value>, f2 = <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6277  FBULE (equal) failed : f0 = <value>, f2 = <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6278  FBO failed: f0 = <value>, f2 = <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6279  FBA failed: f0 = <value>, f2 = <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6280  FBN failed: f0 = <value>, f2 = <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6281  FBU failed.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6282  FBG failed: f0 = <value>, f2 = <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6283  FBUG  failed: f0 = <value>, f2 = <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6284  FBLfailed: f0 = <value>, f2 = <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6285  FBUL failed: f0 = <value>, f2 = <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6286  FBLG failed: f0 = <value>, f2 = <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6287  FBNE failed: f0 = <value>, f2 = <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6288  FBE failed: f0 = <value>, f2 = <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6289  FBUE failed: f0 = <value>, f2 = <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6290  FBGE failed: f0 = <value>, f2 = <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6291  FBUGE failed: f0 = <value>, f2 = <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6292  FBLE failed: f0 = <value>, f2 = <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6293  FBULE failed: f0 = <value>, f2 = <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6294  FBO failed.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6295  fcmpxs failed: Exception did not occur. fsr = <value>
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6296  fcmpxd failed: Exception did not occur. fsr = <value>
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6297  Error: Bus error occured. ftt = <value>
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6298  Error: Bus error did not occur(IEEE exception). ftt =
<value>
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6299  Error: Bus error did not occur(Unfinished exception).
ftt = <value>
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6300  register read/write failed : reg = <value>, expected
/ observed = <value> / <value>
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6301  register test-2 read/write failed : reg = <value>,
expected / observed = <value> / <value>
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6302  register read/write failed : reg = <value>, expected
/ observed = <value> / <value>
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6303  FSR Error: expected / observed = <value> / <value>
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6304  Wrong code is given.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6305   FPU Trap did not occur , i = <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6306   FPU Trap Should not occur but occured.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6307  Bus Error did not occur.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

6308 Did not create correct IEEE exception (Inexact): expected = 1, observed = <value>

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

6309 Did not create correct IEEE exception (Divide By zero) : expected = 2, observed = <value>

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

6310 Did not create correct IEEE exception (Overflow) : expected = 8, observed = <value>

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

6311 Did not create correct IEEE exception (Invalid) : expected = 10, observed = <value>

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

6312 Sun FPU Reliability Test Failed due to segment violation error.

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

6313 Sun FPU Reliability Test Failed due to fpu bus error.

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

6314 Number of failures (out of <value> cases) = <value>

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

6315 malloc(0x<value>) failed :<error_message>

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

6316 Failed <value> precision linpack test.

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6317  DIVTEST <value>: Number of failures (out of <value>
cases) = <value>
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6318  \n                              expected t <value>,
observed <value>\n                              expected x
<value>, observed <value>\n
expected y <value>, observed
<value>\n                               expected z <value>,
observed <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6319  \n                              expected t <value>,
observed <value> \n                              expected x
<value>, observed <value> \n
expected y <value>, observed <value>
\n                            expected z <value>, observed
<value>
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6320  Failed <value> precision Cora test:Check Sum =
<value> Expected <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6321  <value>.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6322  Sun FPU Reliability Test Failed due to floating point
exception error.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
6323  Sun FPU Reliability Test Failed due to illegal
instruction error.
```

Probable_Cause(s):

a. Faulty CPU module.

Recommended_Action(s):

a. Retry test. If the problem persists, contact your Sun Service provider.

```
8000  Failed systest for FPU.

8001  Floating point exception interrupt.
```

11

isdntest 12

Dual Basic Rate ISDN (DBRI) Chip (isdntest)

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

The isdntest is actually a set of several subtests. Three main channels exist within an ISDN: D, B1 and B2. In each of the following subtests, unless otherwise indicated, the D channels are in Basic Rate HDLC data mode, the B1 channels are in 56 kbps HDLC data mode, and the B2 channels are in 64 kbps HDLC data mode. The D channel packet size is 256 bytes, and the B channel packet size is 1024 bytes. The packet count is 10 packets. Each channel runs as an independent thread.

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. Then it activates each interface using the “force activation” capability of DBRI. Each interface is put into local loopback mode (See Figure 12-1). 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.

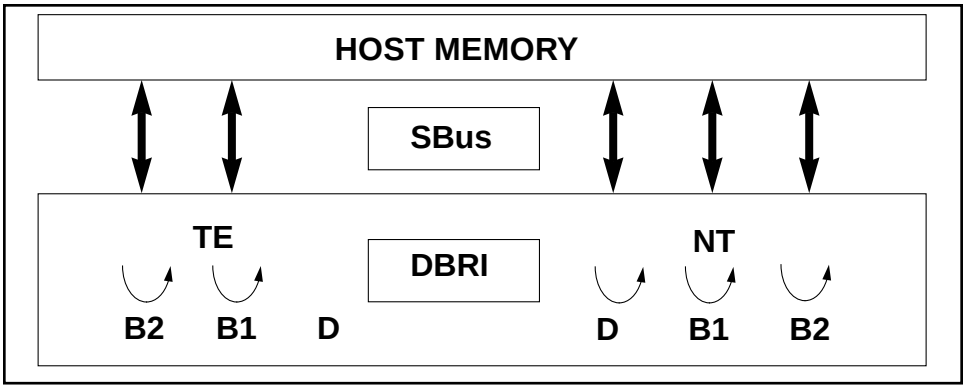


Figure 12-1 `isdntest` Local Loopback Subtest

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 (See Figure 12-2). 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.

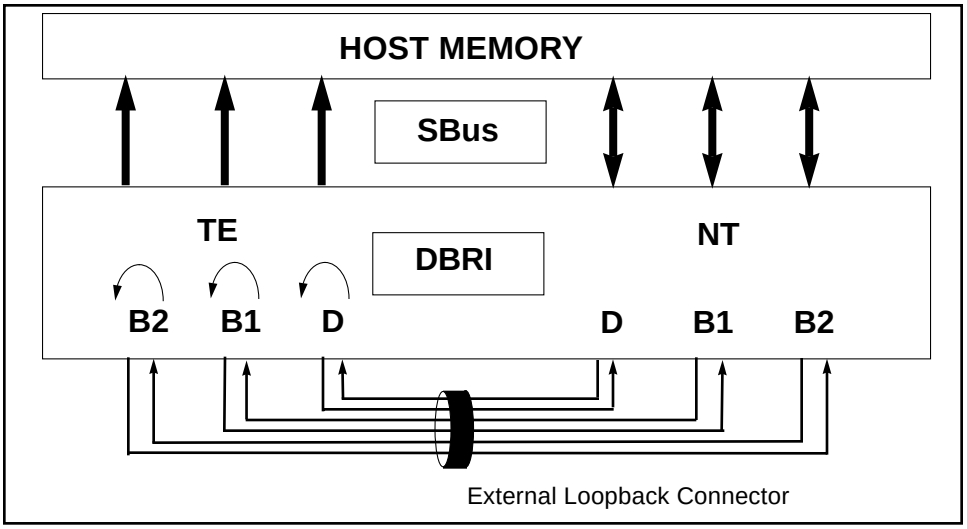


Figure 12-2 `isdntest` Remote Loopback Subtest

Read/write Test

Next, a 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 (See Figure 12-3). 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.

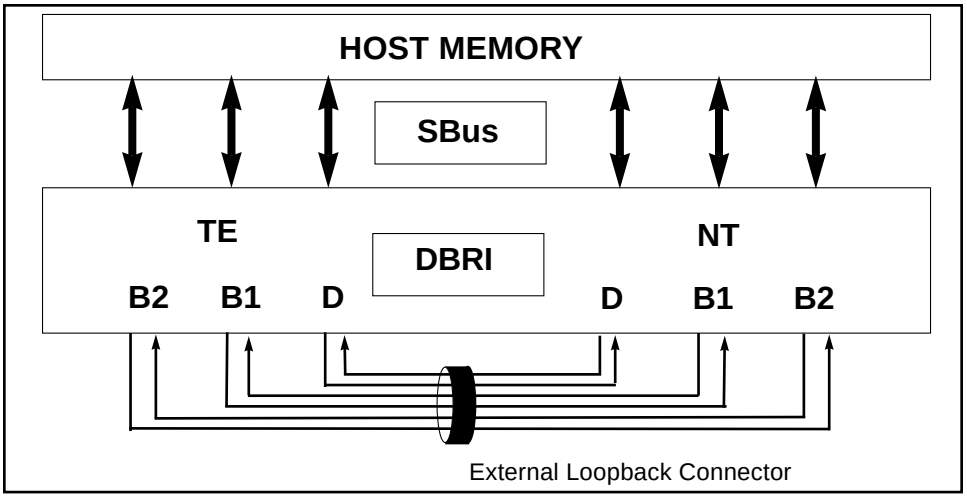


Figure 12-3 `isdntest` Read / Write Subtest

Packet Size Test

The next subtest is a 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 a data path test. Using the ISDN_SET_CHANNEL ioctl, data is routed through a series of short pipe interconnects within DBRI (See Figure 12-4). This subtest requires an NT-to-TE external loopback connector.

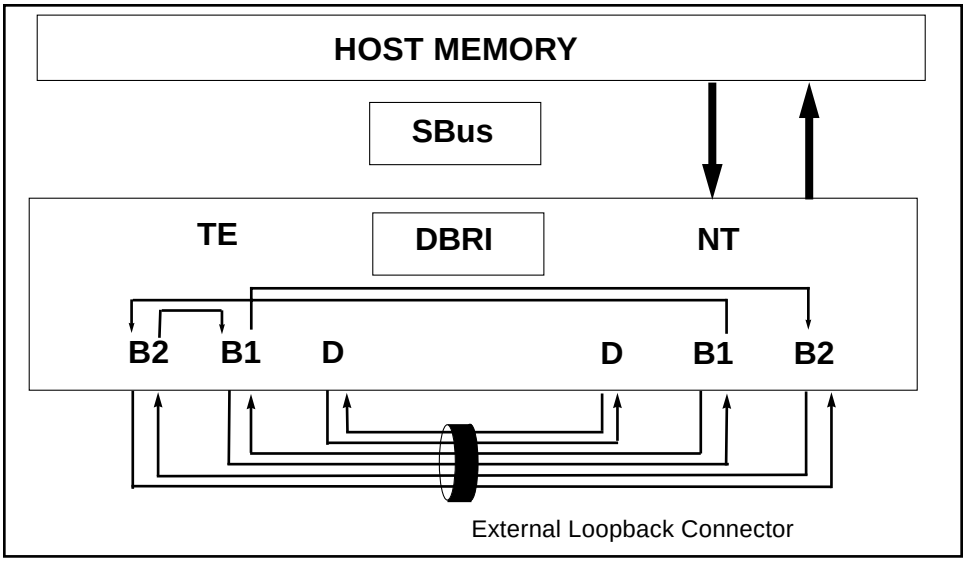


Figure 12-4 `isdntest` Data Path Subtest

`isdntest` *Options*

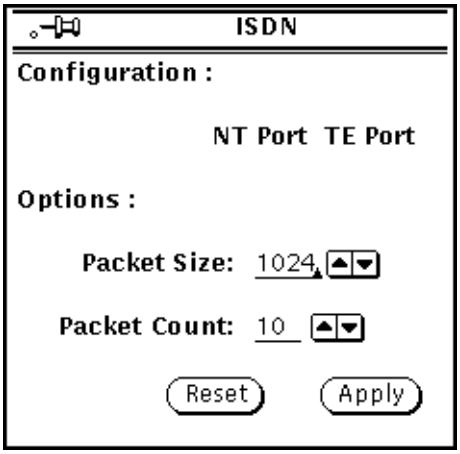


Figure 12-5 `isdntest` Options Menu

Packet Size

Packet size indicates 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

Packet Count indicates how many packets are 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*

Running this test in Connectivity or Online mode restricts testing to the local loopback subtest only.

`isdntest` *Command Line Syntax*

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

Argument	Explanation
`size=`*packet_size*	Specifies the byte size of the B channel packets B channel default size = `1024` bytes D channel default size = 256 bytes B channel maximum packet size = 8186 packets B channel minimum packet size = 1 packet D channel packet size = 256 (except during the packet size test, when it is set to random values between 1 and 256)
`count=`*packet_count*	Indicates how many packets are to be transmitted and received for all channels Default count = 10 packets Maximum packet count = 100 packets

isdntest *Error Messages*

The isdntest error messages are listed below with explanations or probable causes:

1. Unable to open *<device_name>*. (*<error_message>*)

   Device already being used.

2. Write failed for *<device_name>* during *<subtest>*. (*<error_message>*)

3. Return value from write incorrect for *<device_name>* during *<subtest>*.

   Actual: *<number>* Expected: *<number>*

4. Read failed for *<device_name>* during *<subtest>*. (*<error_message>*)

5. Return value from read incorrect for *<device_name>* during *<subtest>*.

   Actual: *<number>* Expected: *<number>*

6. Data miscompare for *<device_name>* during *<subtest>*.

7. Packet *<number>* size *<number>* offset *<number>* contains *<number>*, should be *<number>*.

   Defective loopback cable.

12

leotest 13

*ZX and TZX Graphics Accelerator Test (*leotest*)*

The leotest verifies the Sun Microsystems ZX and TZX Graphics Accelerators with a sequence of subtests. All tests are nondestructive and maintain the system integrity during and after the tests are run.

Caution – Do not run any other application that uses the ZX or TZX accelerator port while running leotest. This combination causes 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.

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

leotest requires approximately 2 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.

This test cannot be run on UltraSPARC systems

13

`leotest` *Options*

By default, SunVTS runs all of the available tests, except the Stereo test. See the test descriptions section below. To avoid excessive test cycle times when testing the ZX or TZX Graphics Accelerators, follow these instructions to ensure that leotest runs once, reports the status as each test routine executes, and then exits:

1. Enable Single Pass on the SunVTS Options menu.
2. Enable Verbose on the SunVTS Options menu.
3. Do not select any other diagnostic tests.

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

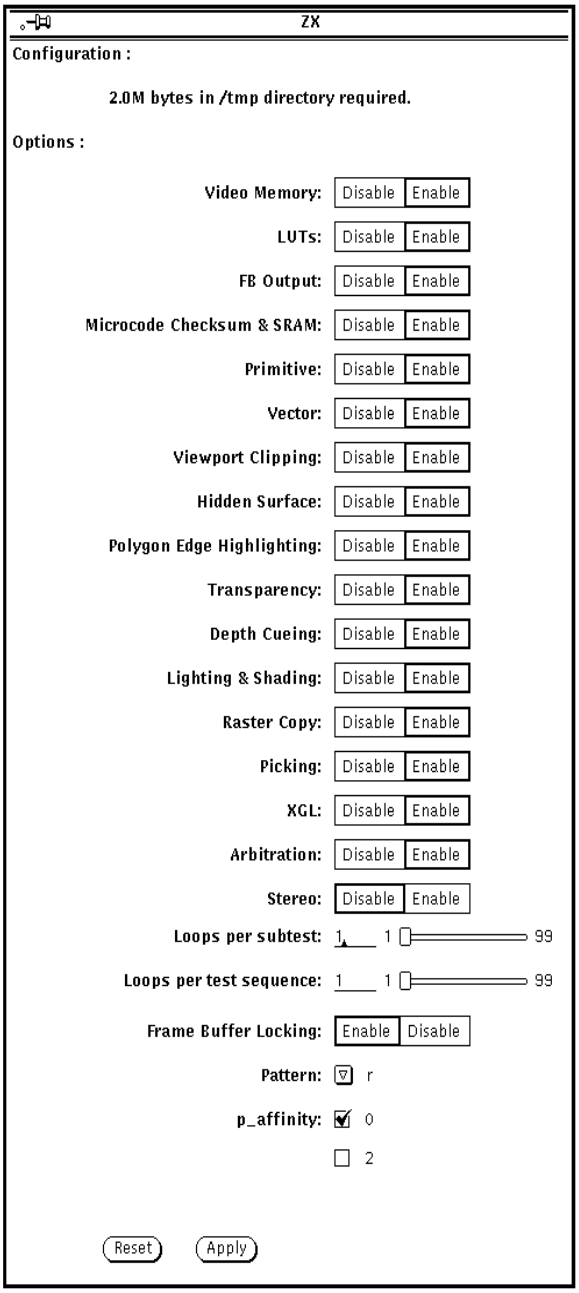


Figure 13-1 `leotest` Option Menu

13

`leotest` *Subtests*

`leotest` is divided into two subtest categories:

- Direct Port tests
- Accelerator Port tests.

Direct Port Tests

The direct ports tests check the non-accelerated portion of the ZX using the following subtests.

Video Memory

The video memory array subtest selects and tests 64 by 64 pixel regions covering all video memory planes, including the 24-bit double-buffered image plane, 8-bit overlay plane, 24-bit depth (Z buffer) plane, and 10-bit window identification (WID) plane. If the subtest detects an error, SunVTS reports the defective plane and location.

LUTs

This subtest performs a nondestructive read-write test on the frame buffer color look-up tables and the WID look-up tables. After the test is complete, the table contents are restored. If this subtest detects a failure, SunVTS reports the location of the failure.

At the beginning of this subtest, red, green, and blue stripes are displayed for visual verification of the digital-to-analog converters (DACs).

Frame Buffer Output

The Frame Buffer Output subtest creates various windows in the Window ID plane, then sets up the look-up tables (LUTs) associated with these windows. This subtest then writes random values to the video memory of these windows. Next, the test verifies the RGB image data of each pixel by triggering the trap registers, reading and generating RGB checksums from these values, and comparing these checksums with known values. This is also a visual cursor test as the cursor is being displayed for each pixel that is being tested.

Note – Moving the mouse cursor during the test can prolong the test and cause failures. For best results, remove the mouse from the mouse pad during this test.

Accelerator Port Tests

The accelerator port test consists of a sequence of subtests that are designed to ensure the ZX Graphics Accelerator integrity at the system level. They take accelerator port transaction files as input. These files contain graphic data that is passed to the ZX Accelerator port in groups of 32-bit words called packets. These packets contain dots, vectors, triangles, and pass-through commands, and are generated in either immediate (programmed I/O) or DMA mode. The ZX SunVTS queries the system software for DMA capability, and if applicable, renders objects in DMA mode for some subtests. For more information on ZX hardware, refer to the *Leo Hardware Reference Manual*.

For verification, after the image is rendered to the frame buffer, each subtest reads the RGB image data from the frame buffer and compares the data against known good images. To save disk space, the good image data is stored in a reduced size (64 times smaller that the normal size), and are stored in the Sun raster file format. The files are stored in the `leotest.data` file in `/opt/SUNWvts/bin`.

Note – The `leotest` subtests verify a frame buffer region of 1152 by 900 pixels, regardless of monitor size.

Microcode SRAM Checksum and Read/Write Selftest

This subtest sends a diagnostic package to the microcode of the floating point transform engine with instructions to reset the accelerator port, run the SRAM selftest, and restart the engine. Then the test verifies each SRAM of all four Leo float chips in the floating point transform engine to make sure they have the same checksum. If not, the test reports an error, based on the information it receives from the microcode. Nondestructive read/write tests are also performed on each SRAM.

Primitive

This subtest renders primitives such as dots and lines, as well as triangle, with different color and shading at each vertex.

Vector

This subtest renders fairly large vector objects with aliased and anti-aliased vectors. This subtest is rendered in DMA mode, when applicable.

Viewport Clipping

This subtest renders and clips an object around and in front of the screen.

Hidden Surface

This subtest renders objects with the Z-buffer-compare attribute turned on.

Polygon Edge Highlighting

This subtest renders an object with the polygon edge attribute turned on. This subtest is rendered in DMA mode, when applicable.

Transparency

This subtest renders a scene with two transparency modes (standalone and alpha blend) in various degrees. This results in a two-pass transparency of the objects in the scene. This subtest is rendered in DMA mode, when applicable.

Depth-Cueing

This subtest renders an object with the depth-cueing attribute turned on.

Lighting and Shading

This subtest renders an object with multiple light sources and Gouraud shading for front and back surfaces. This subtest is rendered in DMA mode, when applicable.

Raster Copy

This subtest renders 32-bit image and various subregions of it, and zooms in on a subregion, using the microcode raster data copy command.

Picking

This subtest has two parts: a pick detect test and a pick echo test.

XGL

The transaction file for this subtest was generated from an XGL program to ensure that the XGL registers are exercised.

Arbitration

This subtest continually renders an object into the accelerator port while a second process performs a read-write test to the WID planes from the direct port on the frame buffer. This subtest simulates conditions where rendering processes and windows operation run concurrently. This subtest is rendered in DMA mode, when applicable.

Stereo (Interactive)

This subtest displays text information in stereo mode. The user verifies proper operation by looking at the screen with stereo glasses and following the instructions displayed.

Loops per subtest

Type the number of times each test should be run before going on to the next test. The default is one loop.

Loops per test sequence

Type the number of times the entire test sequence should be run. Each sequence pass registers one in the SunVTS Pass Count window. The default is one loop.

FB Locking

See the section “Testing Multiple Frame Buffers” in Chapter 3 of the *SunVTS 2.0 User’s Guide* for details.

Pattern

Press MENU to select a data pattern for the direct port tests. The default is “r” for random patterns. You may select a pattern of 0s, 3s, 5s, or 9s. For example, if you choose a pattern of 3s, the value 0x33333333 is written to and read from the frame buffer.

`leotest` *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 mode.

Connectivity Mode

Not applicable for this test.

Online Mode

Not applicable for this test.

Offline Mode

In this mode, `leotest` verifies the ZX and TZX graphics accelerators by running direct port tests and accelerator port tests.

leotest *Command Line Syntax*

/opt/SUNWvts/bin/leotest *standard_arguments* **-o dev=***device_name*,**S=***subtest_number*,**F=***#_of_subtest_loops*,**B=***#_of_test_loops*,**lock=***E(nable)/D(isable)*,**P=***pattern*

Argument	Explanation
dev=*device_name*	*device_name* is the full path name of the device being tested; the default is /dev/fbs/leo0.
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. To run all tests, enter **n=0xFFFF**. **0x 000 001** Direct port—video memory **0x 000 002** Direct port—CLUTs and WID LUTs **0x 000 004** Direct port—Frame buffer output section **0x 000 008** Direct port—SRAM checksum & Read/Write **0x 000 010** Accelerator port—primitives **0x 000 020** Accelerator port—vectors **0x 000 040** Accelerator port—clipping **0x 000 080** Accelerator port—z-buffer **0x 000 100** Accelerator port—polygon edge **0x 000 200** Accelerator port—transparency **0x 000 400** Accelerator port—depth cueing **0x 000 800** Accelerator port—lighting & shading **0x 001 000** Accelerator port—raster copy **0x 002 000** Accelerator port—picking **0x 004 000** Accelerator port—XGL **0x 008 000** Accelerator port—arbitration **0x 010 000** Accelerator port—stereo (interactive)
F=*#_of_subtest_loops*	*#_of_subtest_loops* is the number of loops for each subtest; the default is 1.
B=*#_of_test_loops*	*#_of_test_loops* is the number of loops of each test sequence. The default is 1.

Argument	Explanation *(Continued)*
lock=*E(nable)/D(isable)*	Enables or disables frame buffer locking. See the section "Testing Multiple Frame Buffers" in Chapter 3 of the *SunVTS 2.0 User's Guide* for details.
P=*pattern_number*	Selects a pattern number to be used with the direct ports tests; the default is **r**, for random patterns. You may also choose **0**, **3**, **5**, or **9**.

`leotest` *Command Line Examples*

Here are three examples that illustrate how to run `leotest` from a command line. Make sure to change the directory to `/opt/SUNWvts/bin` before running `leotest` from the command line. `leotest` is hard-wired to look for its data file, `leotest.data`, in `/opt/SUNWvts/bin`.

- A simple accelerator port test, primitive single pass:

```
machine# cd /opt/SUNWvts/bin
machine# leotest -o S=0x10
```

- All direct port tests, five loops of sequence:

```
machine# cd /opt/SUNWvts/bin
machine# leotest -o S=0x7,B=0x5
```

- All subtests (except the interactive tests), two loops of each subtest, four loops of each test sequence:

```
machine# cd /opt/SUNWvts/bin
machine# leotest -o S=0xFFFF,F=2,B=4
```

`leotest` *Error Messages*

Error messages for `leotest` are listed below with explanations and probable causes. In all error conditions, the field replaceable unit (FRU) is the entire FFB. The error messages are listed below, along with explanations and probable cause.

1. `Arbitration test failed.`

   The arbitration test fails, and the cause is given in the message that follows.

2. `Background process wouldn't die. System error.`

   A software error. You may have to reboot the SPARCstation.

3. `Busy wait exceeded <number> loops. Error in the Floating Point Transform section. Rerun Leoconfig.`

   A timeout error condition. There is possibly a problem with the Floating Point Transform section, and the `leoconfig` software (located in `/etc/opt/SUNWleo/bin/leoconfig`) should be executed again. Check the `leoconfig` man pages for more details.

4. `[Plane group name] Byte Access Mode error at x=<number> y=<number>, bank=<number>, expected=<number>, observed=<number>, XOR=<number>.`

   The direct port video memory test found an error at pixel (x,y) in the named plane group. The bank # refers to the corresponding VRAM bank number. Byte/Stencil Access Mode applies to all plane groups that access 8 bits of the frame buffer memory (in other words, the 8-bit image and overlay planes). The test expected to find *exp* but observed *obs*, yielding *xor* when the two values are exclusive `or'd` with each other.

Note – The following error messages are software errors. They are grouped together since the cause of the errors is similar. The reason for the errors is described at the end of the list.

---

5. `Cannot read Window ID look up table from device <device_name>.`

   Check device for existence and/or permissions.

6. `Cannot write Window ID look up table to device <device_name>.`

   Check device for existence and/or permissions.

7. `Cannot post Color LUTs to device <device_name>.`

   Check device for existence and/or permissions.

8. Cannot get monitor mode from device *<device_name>*.

   Check device for existence and/or permissions.

9. Cannot set diagnostic mode from device *<device_name>*.

   Check device for existence and/or permissions.

10. Cannot set monitor mode from device *<device_name>*.

    Check device for existence and/or permissions.

11. Cannot create raster for device *<device_name>*.

    Check device for existence and/or permissions.

12. Cannot create color map for device *<device_name>*.

    Check device for existence and/or permissions.

13. Cannot create context for device *<device_name>*.

    Check device for existence and/or permissions.

14. Cannot create color translation object for device *<device_name>*.

    Check device for existence and/or permissions.

15. Cannot create path object for device *<device_name>*.

    Check device for existence and/or permissions.

16. Cannot create child raster for device *<device_name>*.

    Check device for existence and/or permissions.

17. Cannot create multiple plane group information for device *<device_name>*.

    Check device for existence and/or permissions.

    Software error. The device that you specified (the default is `/dev/fbs/leo0`) may not be available to the test, therefore the above operation cannot be performed on this device. Make sure that you are executing the test on a machine with a ZX, and that you have permission to access it, and that the device is not being used by another application.

18. `Cannot start another process. Software error.`

    Software error. The process table may be full and the SPARCstation may have to be rebooted.

19. `Cannot grab mouse or keyboard because <message>.`

    May need to bring down other graphic software that is currently running same window server.

    Software error. When the FB locking option is selected, `leotest` tries to lock the mouse and keyboard but was not successful. The reason can be one of the following, according the window system software:

    - mouse/keyboard is frozen
    - grab window is not viewable
    - grabbed at invalid time
    - already grabbed by another client

20. `CLUT <number>, index <number>, color <name>, expected <number>, observed <number>, XOR=<number>`

    An error was found in one of the three color look-up tables tested by SunVTS. The error was found in the *n*th CLUT. The index is out of 256 entries in each CLUT. Each CLUT has an 8-bit value for RED, GREEN, and BLUE. The color indicates the 8-bit set in which the error was found. The test expected to find *expected* but received *observed*, yielding *XOR* when the two values are exclusive `or'd` with each other.

21. `Data file <file_name> missing in the current test directory.`

    Software error. ZX SunVTS can't find the data file (`leotest.data`) in the current `/opt/SUNWvts/bin` directory. You may have to reinstall the `SUNWvts` package in the specified directory.

22. `Error in <subtest_name> test.`

    The subtest fails and the cause is given in the message that follows.

23. Error in verifying the <Red/Green/Blue> plane at x=*<number>*, y=*<number>*, bank=*<number>*, expected=*<number>*,observed=*<number>*, XOR=*<number>*.

    Failed accelerator port test. The error is in either the RED, GREEN, or BLUE image plane. The x-y coordinate of the pixel should contain the *expected* value but instead received the *observed* value. This yields *XOR* value, or the bits in error, when the two values are exclusive or'd with each other.

24. Failed to open data file *<file_name>*. Suspect incomplete or incorrect hardware installation. Files may also have been corrupted.

    Software initialization problem. *<file_name>* is the data file that SunVTS can't open.

25. Failed to read data file *<file_name>*. Suspect incomplete or incorrect hardware installation. Files may also have been corrupted.

    Indicates a software initialization problem where *<file_name>* is the data file that SunVTS can't read.

26. Illegal SBus DVMA code = *<number>* addr = 0x%x data= *<number>*.

    Maybe data file is corrupted.

27. Illegal SBus packet, code = *<number>* addr = *<number>* data = *<number>*. Maybe data file is corrupted.

    This error occurs while the subtest is reading the transaction file. It is likely that the data file is corrupted, so the SUNWvts package may need to be reinstalled.

28. Out of Memory.

    No memory left. Increase swap space and/or kill other processes.

29. `Pick Detect misses:<number> lines and/or triangles inside the pickbox and/or <number> lines and triangles outside the pickbox.`

    Failed the Picking accelerator port test. Only the failing component <*Red, Green or Blue*> appears in the message.

30. `Pick Echo failed: *** Error(s) found in` <*Red, Green or Blue*> `components.`

    Failed the Picking accelerator port test. Only the failing component <*Red, Green or Blue*> appears in the message.

31. `Picking: *** Error(s) found in` <*Red, Green or Blue*> `components.`

    Failed the picking accelerator port test. Only the failing component <*Red, Green or Blue*> appears in the message.

32. <*Plane group name*> `Pixel Access Mode error at x=<number> y=<number>, bank=<number>, expected=<number>,observed=<number>, XOR=<number>.`

    The direct port video memory test has found an error at pixel (`x,y`) in the named plane group. Pixel Access Mode applies to all plane groups that access the frame buffer memory 4-bytes at a time. (In other words, all planes except 8-bit planes). The memory for the pixel resides in the given VRAM bank. The test expected to find *expected* but found *observed*, yielding *XOR* when the two values are exclusive or'd with each other.

33. `SRAM checksum mismatch. Float #1 = <number>, Float #2 = <number>, Float #3 = <number>, Float #4 = <number>.`

    All four checksums of the SRAMs in the Floating Point Transform section are not identical. The subtest displays the mismatched checksum from each SRAM of each LeoFloat chip.

34. `SRAM of the LeoFloat [0/1/2/3], number of failures = <number>, first SRAM location = <number>, expected = <number>, observed = <number>, XOR = <number>`

    An error is found in the SRAM test of the LeoFloat chip number 0, 1, 2, or 3. If the number of failures is more than one, the subtest displays the first SRAM location that fails, the expected and observed values in this location, and the bits in error (values of expected and observed are `xor'd` together).

35. `tar never finished. System software problem.`

    Software error. Make sure that the `tar` program is installed correctly on your system. Also, use `df` to see if you have enough disk space left in your `/tmp` directory.

36. `tar <error>`

    Software error. Ensure that the `tar` program is installed correctly on your system. Also, use `df` to see if you have enough disk space left in your `/tmp` directory.

    `tar` failed. Note: A space of approximately 2MB in `/tmp` is required for the test to run correctly.

    Software error. The `tar` program cannot unpack data file to the `/tmp` directory due to limited disk space.

37. `The checksums for <red/green/blue> image data of all pixels don't match, expected = <number>, observed = <number>, XOR = <number>`

    The FB output section subtest prints this message when the checksums of either RED, GREEN, or BLUE image data *observed* from the trap registers in the Video Output test are not the same as the *exp*ected values. It also prints the error messages below if the upper four bits in the trap registers are not set/reset as expected,

38. `The Even Field bit expected to be 0, observed 1.`

39. `The Composite Sync bit expected to be 1, observed 0.`

40. `The Composite Blank field expected to be 1, observed 0.`

41. `The Stereo bit expected to be 1, observed 0.`

42. `The Stereo bit expected to be 1, observed 0.`

    These bits are either not set correctly, or the read is from the wrong location, which indicates there is an error in the Video Output section.

43. `Unable to map <device_name>. Not enough memory.`

    Software error. There is not enough memory to map the addresses for the ZX device; the default is `/dev/fbs/leo0`. You may have to increase swap space or add more memory.

44. `Unable to open device <device_name>. Check device for existence and/or permission.`

    Software error. SunVTS is unable to open the ZX device. Ensure that `/dev/fbs/leo0` exists and that the permissions are correct. There may be a software installation problem in which the ZX software packages need to be reinstalled.

45. `Uncompression of data file failed. Note: A space of approximately 2MB in /tmp is required for the test to run correctly`

    Software error. There is not enough disk space in `/tmp`; about 2 Mbytes is required to uncompress the data file. You may have to remove unneeded files or link `/tmp` to a larger disk partition.

46. `Unknown data file magic number = <number>.`

    Software error. The data file was generated by an older version of software tools. Report this error by filing a bug report or calling the Sun 800 number.

47. `Unsupported 24-bit data length. Maybe data file is corrupted.`

    This error occurs while the subtest is reading the transaction file. Corrupt data is likely, so the `SUNWvts` package may need to be reinstalled.

48. `Unable to open display. Window server not running.`

    Warning message only. This message is displayed when the ZX SunVTS is executed from the command line remotely or if SunVTS is run in TTY mode.

49. `vfork:<error>`

    Software error. An error occurred while trying to fork a child process. Increase swap space, or close other processes.

50. `[PWID/QWID] WLUT: Look up table error at index <number>, expected <number>, observed <number>, XOR <number>.`

    An error was found in either the PWID (Hardware Window ID) or the QWID (Software Window ID) look-up tables. The error was found in the *n*th WLUT. The index is out of 64 entries for PWID or 15 for QWID. The test expected to find *expected* but received *observed*, yielding *XOR* when the two values are exclusive `or'd` with each other, which indicates the bits in error. This error message indicates there is an error in the Video Output section.

lpvitest 14

SBus printer cards have two printer ports: one for SPARCprinters™, and one for any parallel port printer. `lpvitest` verifies the functionality of SPARCprinter ports, and `bpptest` verifies the functionality of bidirectional parallel ports.

Printer Test Hardware and Software Requirements

The SBus Printer Card and device drivers must be installed in order to run `lpvitest` or `bpptest`. A printer must be connected to the SPARCprinter or bidirectional parallel port, and powered-up. If both a SPARCprinter and a parallel port printer are hooked up to the SBus card, you can test both ports simultaneously.

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 printer tray.

The `lpvitest` verifies that your SBus card and SPARCprinter are working properly by attempting to transfer a data pattern from the SBus card to the SPARCprinter and printing the pattern.

If the SBus card and printer are functioning properly, you can see in the SunVTS Status Window that `lpvitest` made a successful pass, and the pattern transmitted to the printer printed correctly.

If this test passes successfully, the SBus DMA circuitry, the SPARCprinter, and the device driver are functioning properly.

`lpvitest` *Options*

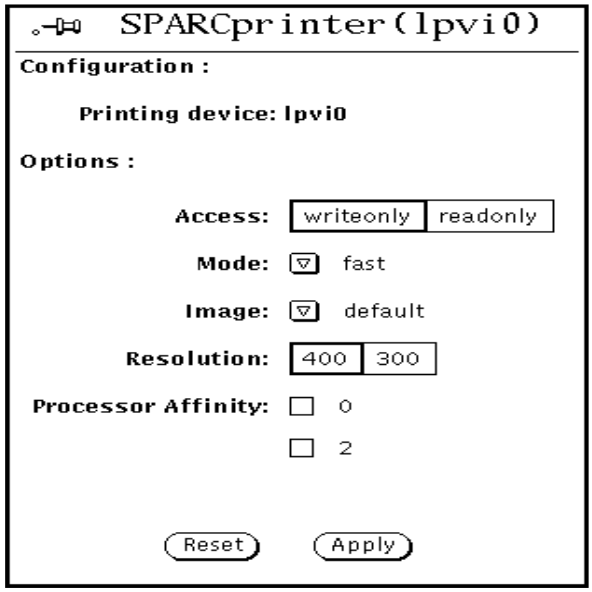


Figure 14-1 `lpvitest` Option Menu

Access

This field is informational only. Writeonly is the only option currently available. This indicates that the only data being transferred is going from the SBus printer card to the SPARCprinter.

Mode

This option allows you to select how often to print the test image. The default setting is Fast; the available settings are:

`Fast` Prints an image every 1 minute.

`Medium` Prints an image every 12 minutes.

`Extended` Prints an image every 30 minutes.

Image

This option lets you choose which test image to print. The settings are:

`Default` A pattern of vertical lines on one page and a checkerboard pattern on another.

`57fonts` An image of the 57 different fonts that the printer supports.

`Userdefined` You can use any rasterfile as a test image. Just place the file in the `/opt/SUNWvts/bin` directory and save it as the file name **`u_image`**.

Resolution

This setting defines the printer resolution of the printed test pattern. The choices are 300 or 400 dots per inch.

Note – Patterns such as the default test pattern are printed at different sizes and at different dpi resolutions. The text in the 57 fonts pattern prints in the same size, using the two different resolutions.

`lpvitest` *Test Mode*

This test only supports the offline mode of testing.

14

lpvitest *Command Line Syntax*

/opt/SUNWvts/bin/lpvitest *standard_arguments* **-o dev=***device_name***,access=***writeonly***|***readonly***,image=***filename***,reso=***resolution***, mode=***mode*

Argument	Explanation	
dev=*device_name*	Specifies the name of the device. This should be of the form **/dev/lpvi**#, where # is the number of the device.	
access=	Choices are write-only or read-only, but just write-only is supported at this time. See Access in Section , “lpvitest Options.”	
image=*filename*	This is the name of the file containing the test image. Possible values are:	
	imagefile	Any file containing user-defined images. Before running SunVTS, copy the raster image file to the /opt/SUNWvts/bin directory and save it with the filename **u_image**. The size of the image is adjusted based on the resolution.
	57fonts	Contains an image of 57 fonts. The size of the image is adjusted, based on the resolution.
	default	The default images print on two pages, one image is a pattern of vertical lines and the other is a checkerboard.
reso=*resolution*	*resolution* is the resolution of the output in dots per inch (DPI). Possible values are **300** and **400**.	
mode=*mode*	*mode* is the print speed mode. This is the rate at which the test image is printed. Possible values are:	
	fast	Prints the test image at 1-minute intervals.
	medium	Prints the test image at 12–minute intervals.
	extended	Prints the test image at 30–minute intervals.

lpvitest *Error Messages*

The lpvitest error messages are listed below, with explanations and probable causes:

6000 Error occurred while attempting to obtain paper size

Probable_Cause(s):

a. <system_error_message>

b. No tray installed

c. Printer out of paper

6001 Error occurred while attempting to fork a child process

Probable_Cause(s):

a. <system_error_message>

6002 image in <name> raster file too big

6003 error=<number>

Probable_Cause(s):

a. <system_error_message>

8000 couldn't open <device_name>

Probable_Cause(s):

a. device does not exist

b. device not configured correctly

8001 ioctl fail on <device_name>

Probable_Cause(s):

a. <system_error_message>

8002 Failed to obtain memory from malloc() system call

Probable_Cause(s):

a. system heavily loaded

```
8003 Couldn't open <file_name>

8004 Error occurred while reading raster file header
<value> for <device_name>
```

Probable_Cause(s):

a. The specified image raster file does not have the proper format

```
8005 Error occurred while loading image file <value> for
<device_name>
```

Probable_Cause(s):

a. The specified image raster file does not have the proper format

```
8006 Error occurred while writing to printer device
```

Probable_Cause(s):

a. <system_error_message>

```
8007 Error encountered while accessing printer device
<device_name>; Paper size: <value>,Counters: <value>, Print
Engine: <value>

8008 Error encountered while accessing printer device
<device_name>; Error code: <value>
```

Probable_Cause(s):

a. Error in main motor

```
8009 Error encountered while accessing printer device
<device_name>; Error code: <value>
```

Probable_Cause(s):

a. ROS out of order

```
8010 Error encountered while accessing printer device
<device_name>; Error code: <value>
```

a. Probable_Cause(s):

b. FUSER out of order

```
8011 Error encountered while accessing printer device
<device_name>; Error code: <value>
```

Probable_Cause(s):

a. XERO failure

```
8012 Error encountered while accessing printer device
<device_name>; Error code: <value>
```

Probable_Cause(s):

a. Interlock open

```
8013 Error encountered while accessing printer device
<device_name>; Error code: <value>
```

Probable_Cause(s):

a. No tray installed

```
8014 Error encountered while accessing printer device
<device_name>; Error code: <value>
```

Probable_Cause(s):

a. No paper exists in selected tray

```
8015 Error encountered while accessing printer device
<device_name>; Error code: <value>
```

Probable_Cause(s):

a. Exit JAM

```
8016 Error encountered while accessing printer device
<device_name>; Error code: <value>
```

Probable_Cause(s):

a. Misfeed JAM

```
8017 Error encountered while accessing printer device
<device_name>; Error code: <value>
```

Probable_Cause(s):

a. Drum cartridge is nearly exhausted

```
8018 Error encountered while accessing printer device
<device_name>; Error code: <value>
```

Probable_Cause(s):

a. Deve module is nearly exhausted

```
8019 Error encountered while accessing printer device
<device_name>; Error code: <value>
```

Probable_Cause(s):

a. No Drum cartridge

```
8020 Error encountered while accessing printer device
<device_name>; Error code: <value>
```

Probable_Cause(s):

a. No Deve cartridge

```
8021 Error encountered while accessing printer device
<device_name>; Error code: <value>
```

Probable_Cause(s):

a. Drum cartridge exhausted

```
8022 Error encountered while accessing printer device
<device_name>; Error code: <value>
```

Probable_Cause(s):

a. Deve cartridge exhausted

8023 Timed out while accessing printer device <device_name>

8024 EIO error

14

mptest 15

Multiprocessor Test (mptest)

The multiprocessor test (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 64 processors can be tested by mptest in a CPU.

The processor mask argument can be sent to mptest during test probing. The mptest verifies that the current processor mask matches what is entered from the command line or from the GUI/TTYUI.

15

`mptest` *Options*

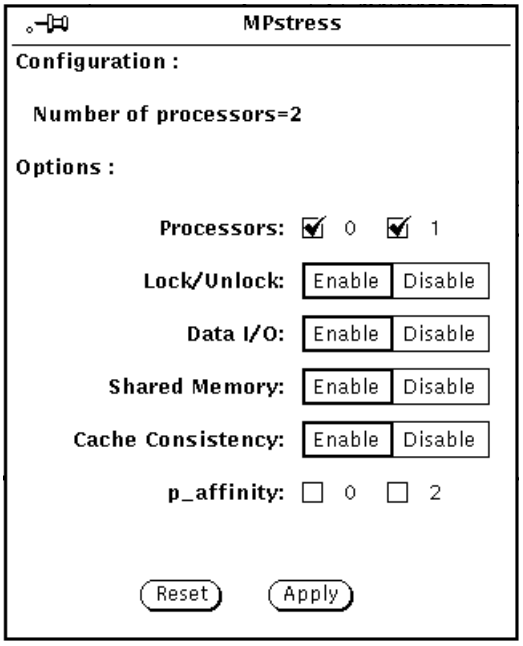


Figure 15-1 `mptest` Option Menu

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

The following options can be run alone or concurrently with other options:

- ***Processors*** lets you specify which processors to test. Click 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 that `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.

- ***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.

15

`mptest` *Test Modes*

Connectivity Mode:

The Connectivity Mode checks the current processors on the system with the original processor mask. An error is reported if they 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 `procesor_bind`.

Online Mode:

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.

Offline Mode:

The offline test mode is the same as Online Mode.

`mptest` *Command Line Syntax*

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

Arguments	Explanation
`NL`	Disable the Lock/Unlock subtest
`ND`	Disable the Data I/O subtest
`NS`	Disable the Shared Memory subtest
`NC`	Disable Cache Consistency subtest
`M`=<*processors to be enabled*	Where 0, 1, 2 are processors
`omask=`<*original mask of processors*>	

`mptest` *Error Messages*

The `mptest` error messages specify problems with specific command-line arguments and explanations or probable causes are listed below:

`6000 mp lock failed`

Probable_Cause(s):

a. Faulty Processor

Recommended_Action(s):

a. If the problem persists, call your authorized Sun service provider.

`6001 All subtests are disabled`

Probable_Cause(s):

a. Parameter error

Recommended_Action(s):

a. At least enable one subtest

`6002 thr_setconcurrency error`

Probable_Cause(s):

a. System load too heavy

`System error`

Recommended_Action(s):

a. Reduce the system load

b. If the problem persists, call your authorized Sun service provider.

`6003 Error in creating threads: <name>`

Probable_Cause(s):

a. System load too heavy

```
System error
```

Recommended_Action(s):

a. **Reduce the system load**

b. **If the problem persists, call your authorized Sun service provider.**

```
6004 thr_join() thread <number> error status: <number>
```

Probable_Cause(s):

a. **System load too heavy**

b. **System error**

Recommended_Action(s):

a. **Reduce the system load**

b. **If the problem persists, call your authorized Sun service provider.**

```
6005 Processor Mask Error: Expect <number> Was <number>
```

Probable_Cause(s):

a. **Some processors are disappeared from processor mask**

Recommended_Action(s):

a. **If the problem persists, call your authorized Sun service provider.**

```
6006 Failed in system call uadmin, errno = <number>,
sys_errlist = <name>
```

Probable_Cause(s):

a. **System load too heavy**

b. **System error**

Recommended_Action(s):

a. **Reduce the system load**

b. **If the problem persists, call your authorized Sun service provider.**

```
6007 Failed in function getnextbitmsk!
```

Probable_Cause(s):

a. System error

Recommended_Action(s):

a. If the problem persists, call your authorized Sun service provider.

```
6008 Processor Mask Error: check mask <number> cpu mask
<number>
```

Probable_Cause(s):

a. Some processors are disappeared from processor mask

Recommended_Action(s):

a. If the problem persists, call your authorized Sun service provider.

```
6009 Failed to get shared memory, err = <name>
```

Probable_Cause(s):

a. System load too heavy

b. System error

Recommended_Action(s):

a. Reduce the system load

b. If the problem persists, call your authorized Sun service provider.

```
6010 Failed to create shared memory, err = <name>
```

Probable_Cause(s):

a. System load too heavy

b. System error

Recommended_Action(s):

a. Reduce the system load

b. **If the problem persists, call your authorized Sun service provider.**

```
6011 Shared Memory Subtest:  Process <number>, Processor
<number>, *curr_add <number> curr_addr <number>, cpu_mask
<number> failed.
```

Probable_Cause(s):

a. **System load too heavy**

b. **System error**

Recommended_Action(s):

a. **Reduce the system load**

b. **If the problem persists, call your authorized Sun service provider.**

```
6014 Failed on read & compare: Processor <number>,
read=<number>, exp=<number>
```

```
6016 Not a MP system!
```

Probable_Cause(s):

a. **Faulty Processor**

b. **Should not run mptest on this system**

```
8000 Failed to open <name>
```

Probable_Cause(s):

a. **Not enough disk space**

b. **Permission error**

c. **System error**

Recommended_Action(s):

a. **Allocate disk space**

b. **Check directory/file permission**

c. **If the problem persists, call your authorized Sun service provider.**

```
8001 Failed to create a page
```

Probable_Cause(s):

a. **System load too heavy**

b. **System error**

Recommended_Action(s):

a. **Reduce the system load**

b. **If the problem persists, call your authorized Sun service provider.**

```
8002 mmap address space to device
```

Probable_Cause(s):

a. **System load too heavy**

b. **System error**

Recommended_Action(s):

a. **Reduce the system load**

b. **If the problem persists, call your authorized Sun service provider.**

```
8003 mlock error.  <name>
```

Probable_Cause(s):

a. **Processor error**

Recommended_Action(s):

a. **Isolate the bad processor.**

```
8005 All subtests are disabled
```

Probable_Cause(s):

a. **Parameter error: None subtest is selected**

Recommended_Action(s):

a. **Re-enter command line correctly: At least select one subtest**

```
8006 Can not run mptest! You need to enable at least two
processors.
```

Probable_Cause(s):

a. Parameter error You have selected less than two processors for testing

Recommended_Action(s):

a. Re-enter command line correctly At least select two processors

nettest 16

*Network Hardware Test (*nettest*)*

The 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), quad Ethernet (QED), fiber optic (fddi), SPARCcluster™ 1 System (em), ATM (sa, ba), and 100 megabits per second Ethernet (be,hme) 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 transport control protocol/interface program (TCP/IP) (ICMP is part of TCP/IP). The target machine must either be configured to respond to ICMP broadcast or to RPC broadcast.

The first thing nettest does is to determine 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*

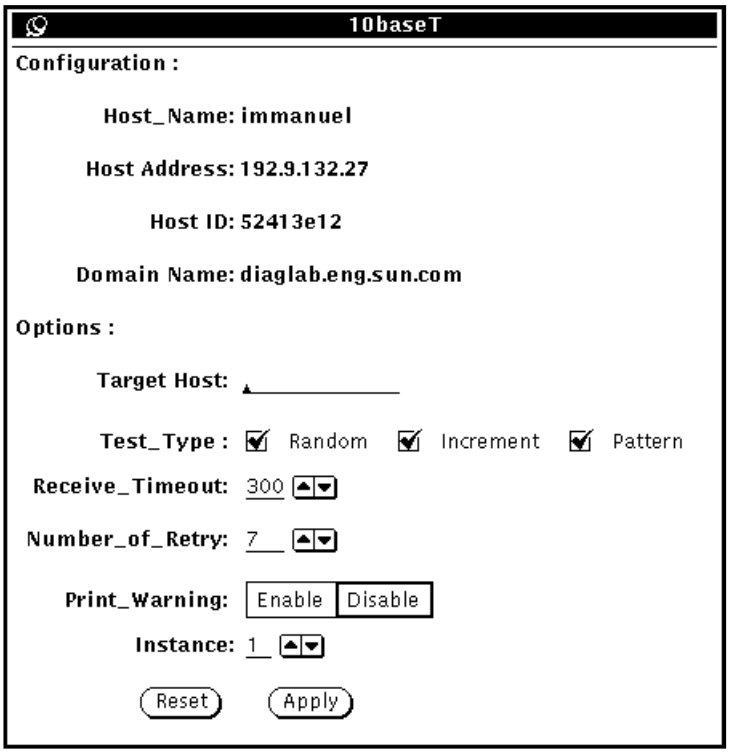


Figure 16-1 `nettest` Option Menu

The Configuration section specifies the Host Name, Host ID, Host Address, and Domain Name of the system being tested.

- ***Target Host field*** 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 via broadcasting. The default setting leaves this field empty.

  Use the check boxes to select either the Random, Increment, or Pattern tests. All three tests are selected as a default.

- ***Receive Timeout field*** default is 120 seconds, but it can be changed. Use a range between 0 - 600 seconds.

- ***Number of Retries field*** default number of retries before flagging an error is three. This field can also be changed: use a range between 0 to 128 retries.
- ***Print Warning field*** is disabled by default. Click on 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.

Connectivity Mode

The nettest checks whether the device is connected or not under this mode. It searches through all the network interfaces for specified device name. The nettest considers the device not connected and test failed if it can not find the match, otherwise it returns device is connected.

Online Mode

Under online mode, nettest is aware that the user's application may be running, it tries to minimize the influence on the application while it is testing. So the nettest uses only random test in online mode. Since the network device can be shared and the system resource is available, applications shall be affected fairly small.

Offline Mode

Under offline mode, the nettest performs all three tests (Random test, Incremental test, and Pattern test) sequentially. It allows user to specify option in such a way that the nettest performs a very stressful test.

nettest *Command Line Syntax*

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

Argument	Explanation
target=*h1***+***h2***+...**	Specifies a list of test targets by hostname or Internet address.
dev=*interface*	Network interface name. The default value is **le0** for Ethernet networks.
test=*type*	Specifies the test type. Type **Random** for only the Random test, **Increment** for the Incremental test, **Pattern** for the Pattern test. The default value is Random+Increment+Pattern for all the tests to be run.
packets=*n*	Number of random / pattern packets. The default is 256.
pattern=*hexn*	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; default is 30 seconds.
timeout=*seconds*	Indicates the number of seconds to wait before time-out; default is 20 seconds.
retry=*n*	Indicates the number of test timeout retries; default is three retries.
warn	When enabled, prints warning messages.

nettest *Error Messages*

The `nettest` error messages are listed below with explanations or probable causes:

```
6000 RPC broadcast failure <error_message>.

6001 Unable to resolve <host_name> ethernet address.

6002 No ICMP echo reply from <host_name>.
```

Probable_Cause(s):

a. system load too heavy

b. No cable connection

c. target machine too busy

Recommended_Action(s):

a. reduce system load or increase timeout time

b. Check cable connection!

c. reduce target machine load

6003 ICMP echo reply incorrect length from <host_name>, exp <data> obs <data>.

6004 ICMP echo reply length <number> from <host_name>, data mismatch at byte <number>, exp <value> obs <value>.

6005 Unable to find any test target.

6006 Unable to find necessary test targets for the ethernet switch.

8000 Unable to open <device_name> <error_message>.

8001 Bad ioctl <name> <error_message>.

8002 Unable to allocate <number> bytes of memory <error_message>.

8003 System call <name> returned <value> <error_message>.

8004 Unable to map network entry for <host_name>.

8005 ATM device <device_name> requires target specification.

8006 Failed to allocate <number> bytes of swap space for configuration.

```
8007 dl_primitive response = <value>

8008 dl_ok_ack: short response

8009 dl_phys_addr_ack: short response

8010 Invalid Argument <string>

8011 Interface <device_name> not configured in system.
```

16

pcsertest 17

*PCMCIA Modem Card Test (*pcsertest*)*

The pcsertest verifies the functionality of the PCMCIA Modem Card. The pcsertest issues a series of commands to the modem that instructs the modem's firmware to run an internal analog loopback diagnostic test. Upon completion, the firmware sends back a three-digit status message indicating whether the test passed or failed.

As an option, the pcsertest tests socket I/O cards. This test writes an 8-Kbyte incrementing data pattern to the I/O card, which is then looped back, read, and verified.

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

17

pcsertest *Options*

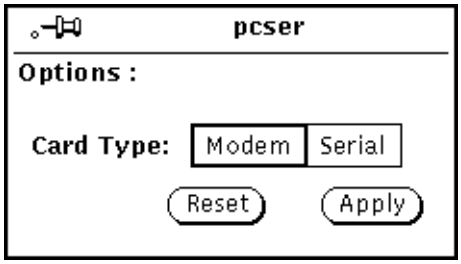


Figure 17-1 pcsertest Option Menu

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.

From the pcsertest Option Window, you can choose to test socket I/O cards. To test a socket card, choose socket on the Card Type switch. If you choose to test a socket I/O card in an empty slot, the test fails.

Any combination of modem and socket I/O cards may be placed in the PCMCIA slots. However, you must select the correct type of card in the Option Window. If you select an incorrect card type, the test fails.

pcsertest *Test Mode*

This test only supports the offline mode of testing.

pcsertest *Command Line Syntax*

/opt/SUNWvts/bin/pcsertest *standard_arguments* **-o dev=***device_name***,type=***card_type*

Argument	Explanation
dev=*device_name*	Specifies the device name (dev=pc0, pc1, etc.)
type=*card_type*	Specifies the card type for the previously specified device name (e.g.,type=serial). You do not need to specify the type if the device is a modem, since modem is the default card type.

pcsertest *Error Messages*

The pcsertest error messages are listed below with explanations and probable causes:

6000 tcgetattr failed for modem card <number> <error message>

6001 tcsetattr failed for modem card <number> <error message>

6002 Write failed while setting modem card <number> <error message>

6003 ioctl TIOCSSOFTCAR failed for serial card <number> <error message>

6004 tcgetattr failed for serial card <number> <error message>

6005 tcsetattr failed for serial card <number> <error message>

6006 Card in slot <number> is not a serial card

6007 Data miscompare error

6008 Modem card <number> returned unknown status

6009 ioctl TIOCMSET failed for serial card <number> <error message>

6010 ioctl TIOCMBIS failed for serial card <number> <error message>

6011 Write to serial card <number> failed. <error message>

6012 Timeout waiting for data from serial card <number>

6013 Read from serial card <number> failed. <error message>

```
6014 Status timeout from modem card <number>

6015 Read from modem card <number> failed. <error message>

6016 Card in slot %d is not a modem card

8000 can't open <device> <error message>

8001 Serial card <number> loopback defective.

8002 Write to modem card <number> failed. <error message>
```

plntest 18

*SPARCstorage Array Controller Test (*plntest*)*

The plntest test checks 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, fiber-channel connection, and the SSA controller board must be working properly to perform this data transfer operation. This is the operation that plntest tries to stress and verify. By stressing this operation, plntest can isolate failures on the SSA disk drives from failures on the SSA Controller board.

Note – disktest transfer data on the SSA disk drives over the same path mentioned above. However, they do not issue these data transfers as quickly as plntest can.

18

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 fiber channel hardware, the SSA fiber 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 SPARC storage array..

plntest *Options*

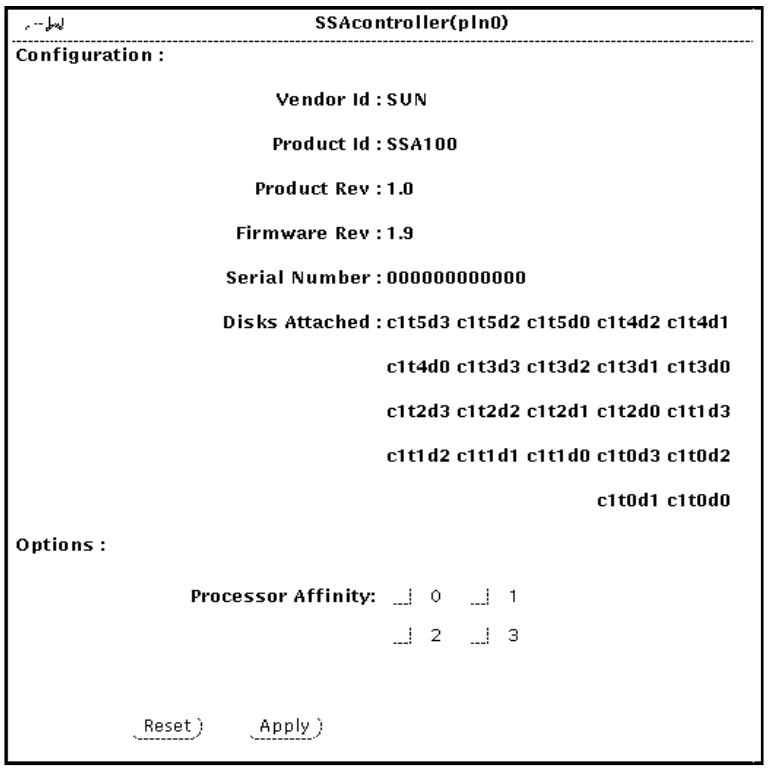


Figure 18-1 plntest Option Menu

- ***Configuration*** lists the names of all the logical disk drives (both single and grouped) that are attached to the SSA controller board.

Note – If no disks are present, the window displays “none” under the Attached Disks heading.

`plntest` is not a scalable test.

`plntest` *Test Modes*

Connectivity Mode

In this mode, `plntest` checks the state of the battery module and the fan module of the SPARCstorage array.

Online Mode

In this mode, `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.

Offline Mode

This mode performs the same type of tests as the Online Mode.

18

plntest *Command Line Syntax*

/opt/SUNWvts/bin/plntest *standard_arguments* **-o ?,dev=***device_name***,x,rw,pattern=***pattern***,IOsize=***buffer_size*

Argument	Explanation
?	Probes the system for valid SSA Controller devices and prints them to screen (see Section , "Probing for SSA Controller Devices.")
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* must be specified for this option to work.
pattern=*pattern*	Specifies the test pattern to be used.
IOsize=*buffer_size*	Specifies the buffer size to be used.

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.

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 comas, 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/ssa`*XX*, where *XX* represents the decimal number of the controller

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

- Manually make a softlink 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`. See the example below:

```
machine# ./plntest "?"

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` *Error Messages*

The `plntest` error messages are listed below, with probable causes and recommended actions. Each error message is preceded by a four-digit number that is unique to this test:

`6010 <device_name> <text1> during <text2>.`

<text1> indicates SCSI sense information for the SCSI command <text2>.

`6011 <device_name> SCSI status returned is <text>,`

Command issued was <text>

`6012 <device_name>: NVRAM Battery is low or has failed.`

Fast writes will be disabled.

Probable Causes:

a. Suspect battery module

Recommended Actions:

a. Consult your authorized Sun service provider.

`8000 <device_name> Failed to open()(<error_message>`

Probable Cause(s):

a. Suspect Fiber Optic Cable

b. Suspect SSA Controller Board

c. Suspect SOC Host Adapter

Recommended Actions:

a. Consult your authorized Sun service provider.

`8010 <device_name> <text1> during <text2>.`

<text1> indicates SCSI sense information for the SCSI command <text2>.

`8011 <device_name> SCSI status returned is <text>,`

Command issued was <text>

`8012 <device_name> Fans have failed.`

Recommended_Actions:

a. Consult your authorized Sun service provider.

pmem 19

*Physical Memory Test (*pmem*)*

This test checks the physical memory of the system. The pmem test 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.

pmem *Options*

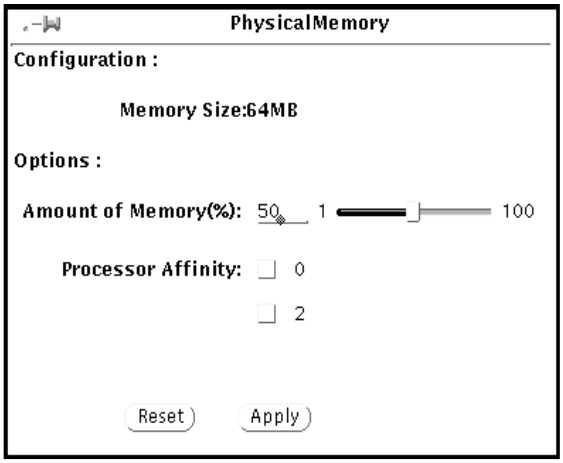


Figure 19-1 pmem Option Menu

The amount of memory shown in the configuration field is the total physical memory probed by the SunVTS kernel. It reflects the amount of physical memory found, rounded up to the nearest megabyte by any fraction. The options to the test allows the user to choose the amount of memory to be read.

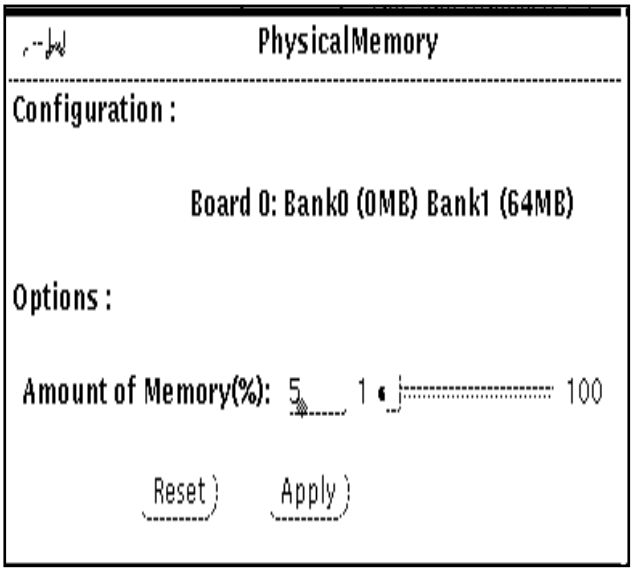


Figure 19-2 UltraSPARC Server Option Menu

In the case of UltraSPARC servers, the configuration field lists the banks of memory on each cpu / memory board separately. When physical mapping is used pmem test is registered for each cpu / memory board separately.

pmem *Test Modes*

Connectivity Mode

In this mode, 1 per cent of the memory is read. For 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.

Online Mode

A variable amount of memory is read in online mode; the default is 50 per cent. For UltraSPARC servers, the ECC error reporting is the same as for the connectivity mode.

Offline Mode

In offline mode, the amount of memory to be read can vary; the default is 50 per cent. For UltraSPARC servers, this test mode reports all ECC errors that have occurred since the system was brought online.

pmem *Command Line Syntax*

/opt/SUNWvts/bin/pmem *standard_arguments* **[-p=n] [-i=n] [-w=n] [size=n] [bdinfo=n]**

Argument	Explanation
size=*number*	Specifies the percentage of memory to be read
bdinfo=*number*	For UltraSPARC servers, indicates the board number information. For example, if board 0 and board 5 have memory and you want the test to read the memory on these two boards, then **bdinfo**=*33 (2**5 + 2**0)*

pmem *Error Messages*

The pmem error messages are listed below with probable causes and recommended actions. Each error message is preceded by a four-digit number that is unique to this test:

`6003: <number> intermittent errors on <text>.`

<text> would indicate the SIMM locations.

`6004: Symbol <text> not found in kernel image.`

<text> will indicate the name of the symbol not found in the kernel image.

`6005: <number> persistent errors on <text> SIMM(s).`

<text> would indicate the SIMM locations.

Probable_Cause(s):

a. <simm_location> SIMM(s).

Reccomended_Action(s):

a. Call your service representative.

```
8000  kvm_open() failed: <error_message>.
```

```
8001  sysinfo(SI_PLATFORM) failed: <error_message>.
```

```
8002  Symbol <text> not found in kernel image.
```

<text> will indicate the name of the symbol not found in the kernel image.

```
8003  Couldn't open file /dev/mem: <error_message>.
```

```
8004 Unable to read pointer to memlist structure.
```

```
8005 Unable to read "size" and "address" in memlist.
```

```
8006 llseek() to address <address> failed
(<error_message>).
```

```
8007 read() at address <address> failed (<error_message>).
```

```
8008 sysconf(_SC_NPROCESSORS_ONLN) failed: <error_message>.
```

```
8009 llseek to address <address> failed (<error_message>).
```

```
8010 llseek to address <address> [<text>] failed
(<error_message>).
```

Note – This message is seen on UltraSPARC servers only. The <text> indicates to which board and bank the physical address <address> maps.

```
8011 read() at address <address> [<text>] failed
(<error_message>).
```

Note – This message is seen on UltraSPARC servers only. The <text> indicates to which board and bank the physical address <address> maps.

19

pstest 20

*Prestoserve Test (*pstest*)*

The Prestoserve™ is an NFS accelerator. It reduces the frequency of disk I/O access by caching the written data blocks in nonvolatile memory. Prestoserve then flushes the cached data to disk asynchronously, as necessary.

The `pstest` verifies the Prestoserve accelerator's functionality with the following three checks:

Board Battery Check

To ensure proper battery power level, the test runs this check before running the other two checks. If it finds a bad battery, it exits `pstest` immediately with a fatal error message.

Board Memory Check

This check maps the entire board memory to a process address space and locks the board to prevent multiple accesses. The test then travels through the mapped address spaces sequentially, doing a char, short, and long comparison on each space. The `pstest` executes this check twice.

Board Performance and File I/O Access Check

This check exercises only synchronous read/write access. `pstest` writes data equal to the amount of on-board memory to the memory cache and reads the data back for comparison. The time taken to write that data is measured twice:

the first time with the Prestoserve accelerator disabled, the second time with Prestoserve enabled. The first value is divided by the second to get the performance ratio. If the ration is less than 1.5 on three passes of the test, the Prestoserve board may or may not have a problem.

Since the Prestoserve accelerator speeds up the `/opt` partition during testing and this partition may be mounted remotely, there may or may not be a problem with the Prestoserve board itself; rather, a network performance problem could be the problem. In either case, a warning message is displayed if the performance ratio is less than 1.5.

Caution – To insure that consistent results are obtained, run as many concurrent tests as possible when a Prestoserve product has been installed. Tests are selected from the Option menu. The default is 2; the maximum is 10. However, do not enable `pstest` and `kmem` at the same time. Running these tests together causes SunVTS to report erroneous errors.

`pstest` *Options*

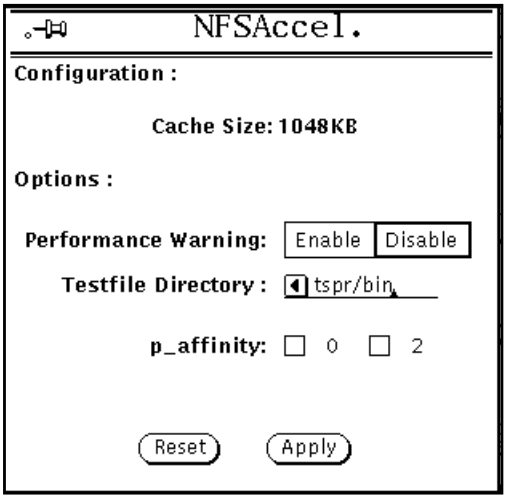


Figure 20-1 `pstest` Option Menu

pstest *Test Modes*

Connectivity Mode

Not available for pstest.

Online Mode

Not available for pstest.

Offline Mode

The test executes the Board Battery Check subtest, Board Memory Check subtest, and Board Performance and File I/O Access Check subtest.

pstest *Command Line Syntax*

/opt/SUNWvts/bin/pstest *standard_argument* **-o e,l,f=***testfile_directory*

Argument	Explanation
e	Enables performance warning, which displays a warning message if the performance ratio is less than 1.5.
l	Enables long memory test.
f=*testfile_directory*	Specifies the directory of a test file.

pstest *Error Messages*

The pstest error messages are listed below with explanations or probable causes:

```
6001  Checking prstatus failed: <error_message>

6002  Failed to turn prestoserve DOWN: <error_message>

6003  File I/O test failed with low performance ratio:
<number>
```

Recommended_Action(s):

a. Please check that prestoserve is configured for /tmp.

```
6004  byte compare: location = <address>, observed =
<number>,expected = <number>

6005  word compare: location = <address>, observed =
<number>, expected =<number>



6006  (data type) long compare: location = <address>,
observed = <number>, expected = <number>

6007   lseek() failed: <error_message>

6008   NVRAM write failed: <error_message>

6009   NVRAM read failed: <error_message>

6010   Failed to get status of Prestoserve failed:
<error_message>

6011   Failed to reinitialize Prestoserve: <error_message>

6012   Failed to turn prestoserve UP: <error_message>

6013   Failed to munmap prestoserve: <error_message>

6014    File write failed: <error_message>

6015    File read failed: <error_message>

6016   Data compare failed.

6017   Failed to restore prestoserve state: %s

6018   Prestoserve test is applicable in Offline mode only.

8001   Failed to open prestoserve board: <error_message>
```

Probable_Cause(s):

a. Prestoserve hardware not installed

b. NVSIMM/NVRAM SBus Board not seated right.

c. Prestoserve software not installed.

d. Not in superuser mode

```
8002  Prestoserve is in the ERROR state.
```

Probable_Cause(s):

a. Errors have occurred on a disk drive.

```
8003  Some batteries are not good or not enabled.

8004  memory_check (pass <pass_id>) failed.

8005   fileio_check (pass <pass_id>) failed.

8006    File I/O test failed (Prestoserve down)

8007    File I/O test failed (Prestoserve up)

8008   Open %s also failed: %s. Please enter test file
directory.

8009    PRESTOSERVE TESTS FAILED!
```

20

SunVideo Test (`rtvctest`*)*

The `rtvctest` verifies the functionality of the SunVideo™ SBus card. SunVideo technology captures and compresses video input in real-time, making it possible to have realtime video conferencing over standard Ethernet networks.

The `rtvctest` is available in SunVTS2.0 as a customtest. To enable the test to show up on the SunVTS user interface, you must:

1. edit the `.customtest_OtherDevices` file in the SunVTS bin directory
2. uncomment the line starting with the label SunVideo.

The `rtvctest` was originally written for Sundiag and the messages from the test are displayed on the SunVTS console within a SunVTS INFO message wrapper.

The `rtvctest` is available under the group `OtherDevices`, and is only available in offline mode.

`rtvctest` *Subtests*

The `rtvctest` is divided into four sub-tests: PROMCheck, memory, Jalapeno, and CL4000. The PROMCheck sub-test verifies the SunVideo card's programmable read only memory. The memory test verifies all of the memory on the card, including the 2 Mbytes of memory on the CL4000 compression

engine and the memory on the Jalapeno application-specific integrated circuit (ASIC). The Jalapeno sub-test verifies the interface logic between the SBus, A/D conversion chips, and the CL4000 compression engine. The CL400 sub-test verifies that the compression engine ASIC is able to compress digitized video data from the A/D chips and send this data to the SBus, through the Jalapeno ASIC.

The `rtvctest` is composed of 49 verification test modules. Table 21-1 lists these modules and their associated test sequence numbers.

Table 21-1 `rtvctest` Verification Modules

SunVideo Verification Module Name	Test Sequence Number
RTVC SUNDIAG Start	0
RTVC Checksum	1
RTVC Jalapeno SMEM	2
RTVC CL4000 DMEM	3
RTVC Jalapeno SBus Interrupt Mask	4
RTVC CL4000 Interrupt Mask	5
RTVC DVMA Control Register	6
RTVC DVMA Transfer Size Counter	7
RTVC DVMA Memory Address Counter	8
RTVC DVMA Virtual Memory Address Counter	9
RTVC DVMA Slave SBus Rerun Register	10
RTVC IIC Control Register	11
RTVC IIC Data Register	12
RTVC Video DMA Control Register	13
RTVC Video DMA Transfer Size Counter	14
RTVC Video DMA Memory Address Counter	15
RTVC User Interrupt 0	16
RTVC User Interrupt 1	17
RTVC User Interrupt 2	18
RTVC User Interrupt 3	19

Table 21-1 `rtvctest` Verification Modules *(Continued)*

SunVideo Verification Module Name	Test Sequence Number
RTVC Video Control and Status Register	20
RTVC Video Control Field Line Interrupt 1	21
RTVC Video Control Field Line Interrupt 2	22
RTVC Video Scan Line Mask Registers	23
RTVC Video Input Format Type	25
RTVC Video Horizontal Lock	26
RTVC Video Even Odd Field	27
RTVC CL4000 Host Control	28
RTVC CL4000 Host Lock	29
RTVC CL4000 Video Port A Control	30
RTVC CL4000 Video Port B Control	31
RTVC CL4000 Video Port A FIFO	32
RTVC CL4000 Video Port B FIFO	33
RTVC CL4000 Address Memory Registers	34
RTVC CL4000 Instruction Memory Access Registers	35
RTVC Time Stamp Register	36
RTVC CL4000 Register Memory	37
RTVC CL4000 Scratch Memory	38
RTVC CL4000 DMA Mode	39
RTVC CL4000 Motion Estimation Registers	40
RTVC CL4000 PSW	41
RTVC CL4000 Variable Length Coder Registers	42
RTVC CL4000 Channel Memory Registers	43
RTVC CL4000 CPU Control	44
RTVC CL4000 Multiply Control	45
RTVC CL4000 DMA Interrupt Control	46
RTVC CL4000 Block Transfer Mode	47

Table 21-1 `rtvctest` Verification Modules *(Continued)*

SunVideo Verification Module Name	Test Sequence Number
RTVC CL4000 Accumulator MSB	48
RTVC CL4000 JPC Field	49
RTVC SUNDIAG Finish	50

`rtvctest` *Options*

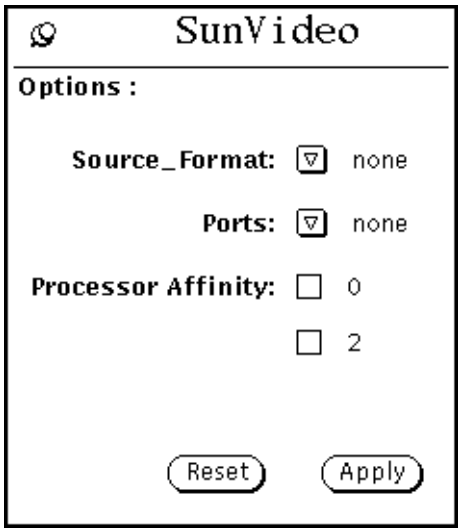


Figure 21-1 `rtvctest` Option Window

You may test the SunVideo card without any video device (camera, video disc player, or VCR) connected to a input port. However, if you connect a video device to the card, then you must state the format of the video source and the input port which the device is connected.

Note – If you do not state both the video source and the input port, the `rtvctest` prints an error and terminates testing.

Source Format

Select the format of the video source used for testing. You may select from the following sources:

Video Source	Definition
None	No video source.
NTSC	A National Television System Committee (NTSC) video source, which is the video standard in the United States and Japan.
PAL	A phase alternate line video source, which is the video standard in Europe.

Ports

If you have selected a video source for testing, then you must state which input port you have used to connect the source to the SunVideo card. You may select from the following ports:

Input Port	Definition
None	No input port used.
CV1	The Composite Video Input Port 1 (RCA type input).
CV2	The Composite Video Input Port 2 (RCA type input).
SVHS	The Super VHS input port.

21

`rtvctest` *Command Line Syntax*

`/opt/SUNWvts/bin/rtvctest [fmt=none|ntsc|pal] [port=none|cv1|cv2|svhs] dev=`*rtvc_device***

Arguments	
`fmt`	Select the format of the video source used for testing. Select either **`ntsc`** (video standard in the United States and Japan) or **`pal`** (video standard in Europe).
`port`	If you have selected a video source for testing, then you must state which input port you have used to connect the source to the SunVideo card. You may select from the following ports: **`cv1`** Composite video input port 1 (RCA type) **`cv2`** Composite video input port 2 (RCA type) **`svhs`** Super VHS port
dev	You are required to state the SunVideo device when running the rtvctest from the command line. Replace *rtvc_device* with **`rtvc`***n*, where *n* is the device number (from 0 to 31) of the SunVideo card under test.

`rtvctest` *Error Messages*

SunVideo Error Message #10

<message>

Where:

<message> is one of the following:

```
Missing REQUIRED argument D=rtvc[0..31]
```

SunVideo Error Message #20

<message> `Error` # <number>

Where:

<message> is one of the following:

```
ioctl RTVC_CMD_RESET fault
ioctl RTVC_CMD_SET_VIDEO fault
ioctl RTVC_CMD_GET_VIDEO fault
open /dev/rtvc fault
close /dev/rtvc fault
open /dev/rtvcctl fault
close /dev/rtvcctl fault
Unknown Jalapeno Version
```

<number> is one of the following:

```
ioctl error
module version
```

SunVideo Error Message #30

<message> <name> Error # <number>

Where:

<message> is one of the following:

memory map fault for
memory unmap fault for
selected /dev/rtvc and /dev/rtvcc1 not available

<name> is one of the following:

prom
sram
buses
reset
dmem
rtvc

<number> is the error number

SunVideo Error Message #40

RTVC Fault Detected via module <name>
Physical Address <value> Expected Value <value> Actual Value <value>
RTVC Sundiag Module number <number>
Location <location>
Message: <message(s)>

Where:

<name> is the name of the module under test

<value> are the values associated with physical address, expected value, actual value

<number> is the Sundiag Module under test

<location> is the U location of the faulty component

<message(s)> are any additional messages

SunVideo Error Message #41

`RTVC Fault Detected via module` <name>

Where:

<name> is the name of the module under test

SunVideo Error Message #42

<message> <value1> Actual <value2>

Where:

<message> is one of the following:

```
Video Format Expected
Video Port Expected
```

<value1> is the expected value

<value2> is the actual value

SunVideo Error Message #50

<message> <number>

Where:

<message> is one of the following:

```
Unknown RTVC CL4000 test module
Unknown RTVC Jalapeno test module
Unknown RTVC test module
Unknown RTVC memory test module
Unknown Start Bit
```

<number> is the module number or start bit location

21

spdtest 22

*NeWSprinter Test (*spdtest*)*

The spdtest is a two part test that checks the printer support hardware. The first part is a register test that checks the NeWSprinter™ 20 SBus printer card's internal functions. The second part is a printing test that checks the interaction between the printer and the print server, as well as the printer's capabilities.

22

`spdtest` *Options*

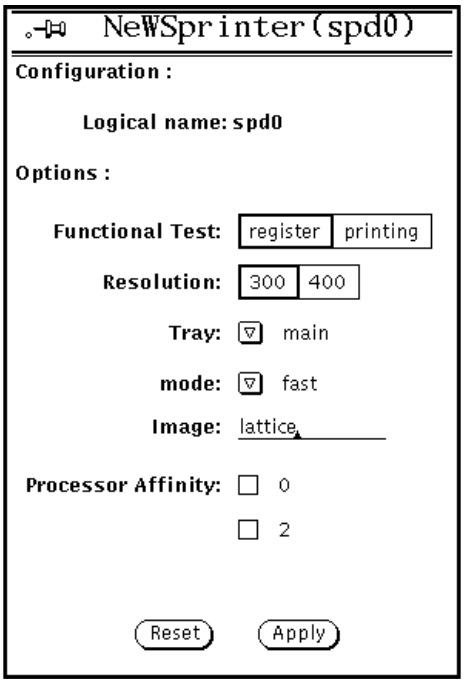


Figure 22-1 `spdtest` (NeWSprinter) Option Menu

Functional Test

The Functional test is composed of two subtests:

- The Register test checks the NeWSprinter 20 SBus printer card.
- The Printing subtest tests the printer's capabilities.

The default setting is Register.

Resolution

This exclusive setting defines the printer resolution of the printer test pattern. You can choose either 300 or 400 dots per inch (dpi). The default setting is 300 dpi.

Tray

From the Tray menu, you can choose from five different types of paper cassettes:

- Main Tray (or Tray 1) is the top cassette that feeds paper to the printer.
- Tray 2 is the bottom cassette that feeds paper to the printer.
- Tray 3 is an optional tray.
- Manual Tray is a manual feed tray connected to the rear of the printer. This tray is used to hand-feed print media that cannot be accommodated by the paper cassettes.
- Auxiliary Feeder is optionally installed in place of the manual feed tray. The Feeder stores and automatically feeds print media that would otherwise require the manual feed.

The default setting is the Main tray.

Mode

From the Mode menu, you can set the interval at which images are printed. The choices are:

- `Fast` prints an image every 10 seconds.
- `Medium` prints an image every 12 minutes.
- `Extended` prints an image every 30 minutes.

The default setting is `Fast`.

Image

This option lets you enter the file name of the test image to print.

`spdtest` *Test Mode*

This test only supports the offline test mode.

22

`spdtest` *Command Line Syntax*

`/opt/SUNWvts/bin/spdtest` *standard_arguments* **`-o dev=`***device_name*`,`
`func=`*function*`,reso=`*resolution*`,tray=`*tray*`,mode=`*mode*`,image=`*image*

Argument	Explanation
`dev=`*device_name*	Specifies the full path name of the device.
`func=`*function*	Represents one of the following functional tests: **`register`** = Register test **`printing`** = Printing test
`reso=`*resolution*	Specifies a number that indicates print resolution: **`300`** = 300 dpi **`400`** = 400 dpi
`tray=`*tray*	Specifies a number that represents the printer cassette type: **`main`** = Main tray (Tray 1) **`tray2`** = Tray 2 **`tray3`** = Tray 3 (optional) **`manual`** = Manual Tray **`aux`** = Auxiliary feeder (optional)
`mode=`*mode*	Specifies a number that represents how often to print the test image: **`fast`** = Fast (Prints an image every 10 seconds) **`medium`** = Medium (Prints an image every 12 minutes) **`extended`** = Extended (Prints an image every 30 minutes)
`image=`*image*	Specifies a rasterfile name.

`spdtest` *Error Messages*

The `spdtest` error messages are listed below with explanations and probable causes:

```
6000 Test function is not selected
```

Recommended_Action(s):

a. Select from register and printing for the Functional Test (func) parameter

```
6001 Error occured while driver was runing self diagnostics
on spd device
```

Probable_Cause(s):

a. Initialization Error

Recommended_Action(s):

a. Retry. If failure persists, contact service representative

```
6002 Error occured while driver was runing self diagnostics
on spd device
```

Probable_Cause(s):

a. Read/Write Error

Recommended_Action(s):

a. Retry. If failure persists, contact service representative

```
6003 Error occured while driver was runing self diagnostics
on spd device
```

Probable_Cause(s):

a. Loopback Error

Recommended_Action(s):

a. Retry. If failure persists, contact service representative

```
6004 Error occured while printing raster image
```

Probable_Cause(s):

a. Printer RAM failure

Recommended_Action(s):

a. Retry. If problem persists, contact service representative

```
6005 Error occured while printing raster image
```

Probable_Cause(s):

a. Printer motor failure

Recommended_Action(s):

a. Retry. If problem persists, contact service representative

```
6006 Error occured while printing raster image
```

Probable_Cause(s):

a. Printer ROS failure

Recommended_Action(s):

a. If problem persists, contact service representative

```
6007 Error occured while printing raster image
```

Probable_Cause(s):

a. Printer fuser failure

Recommended_Action(s):

a. Replace the fuser unit

```
6008 Error occured while printing raster image
```

Probable_Cause(s):

a. Printer XERO failure

Recommended_Action(s):

a. Properly install or replace toner kit

```
6009 Error occured while printing raster image
```

Probable_Cause(s):

a. No toner kit

Recommended_Action(s):

a. Install a new toner kit

```
6010 Error occured while printing raster image
```

Probable_Cause(s):

a. Toner end of life

Recommended_Action(s):

a. Replace toner kit

```
6011 Error occured while printing raster image
```

Probable_Cause(s):

a. Printer cover open

Recommended_Action(s):

a. Close printer cover

```
6012 Error occured while printing raster image
```

Probable_Cause(s):

a. Paper jam

Recommended_Action(s):

a. Clear paper jam

```
6013 Error occured while printing raster image
```

Probable_Cause(s):

a. No cassette in the selected tray

```
6014 Error occured while printing raster image
```

Probable_Cause(s):

a. No paper in the selected tray

```
6015 Error occured while printing raster image
```

Probable_Cause(s):

a. No paper in the manual tray

```
6023 Error occured while printing raster image
```

Probable_Cause(s):

a. Late error

Recommended_Action(s):

a. Contact service representative

```
6024 Error occured while printing raster image
```

Probable_Cause(s):

a. Serial buffer overrun

Recommended_Action(s):

a. Contact service repreentative

```
6025 Error occured while printing raster image
```

Probable_Cause(s):

a. Serial error

Recommended_Action(s):

a. Contact service representative

6026 Error occured while printing raster image, in setting tray or resolution

6027 Error in setting tray or resolution

6028 Error occured while printing raster image

Probable_Cause(s):

a. Specified file is not a raster file

6029 Error occured while printing raster image

Probable_Cause(s):

a. Specified file is not a raster file

b. Not enough memory

6030 Error occured while printing raster image

Probable_Cause(s):

a. File error: raster width does not fall on 32 bit boundary

6031 Failure to write raster header to file

6032 Failure to write raster image to file

8000 <device_name> not found

Probable_Cause(s):

a. Device does not exist on system under test

8001 <device_name> could'nt be opened

Probable_Cause(s):

a. The device is not configured correctly

8002 Image file could'nt be opened

Probable_Cause(s):

a. The specified image file does not exist

```
8003 Invalid parameter(s)
```

Recommended_Action(s):

a. Please refer to the routine usage

```
8004 Can't create raster file
```

Recommended_Action(s):

a. Check the directory permission, user status, etc.

```
8005 Error occured while printing raster image
```

Probable_Cause(s):

a. Printer power off

```
8006 Cannot get memory
```

spif 23

*Serial Parallel Controller Test (*spif*)*

The spif test accesses card components such as the cd-180 and ppc2 chips, and the serial and parallel ports via the Serial Parallel Controller device driver.

spif *Hardware Requirements*

Before running the SunVTS system exerciser, make sure you install the cards to be tested and the device driver. 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 – The spif test 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 No. 540-2007)
- 96-pin loopback plugs (Part No. 370-1366)
- 25-pin serial loopback plugs (Part No. 540-1558)

- RS-232 serial cables (Part No. 530-1685)
- TTY terminal

`spif` *Options*

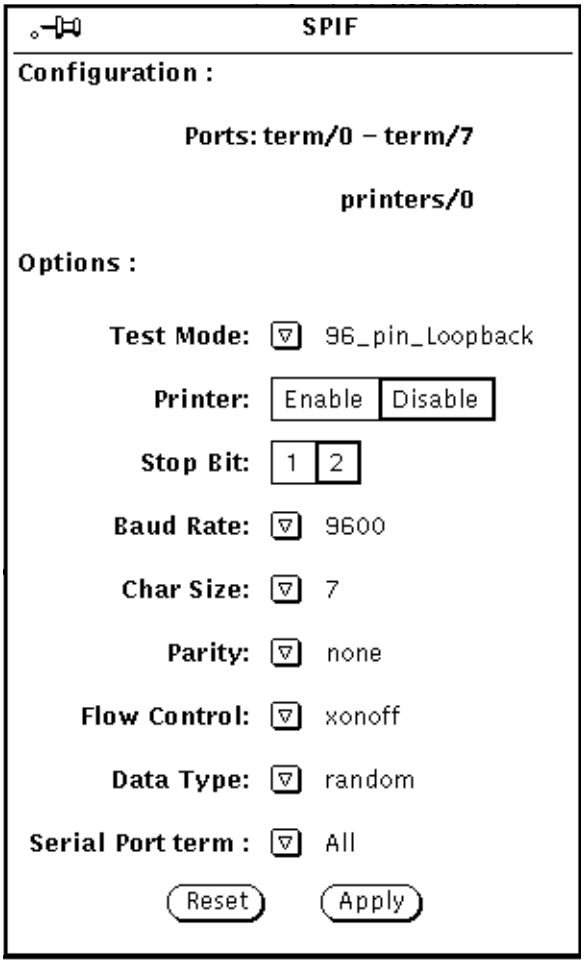


Figure 23-1 `spif` Option Menu

The Configuration section of the option menu displays the serial ports available for the SPC/S board. The available ports are:

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 `spif` test options are described as follows:

- ***96-pin Loopback (LB)*** test provides data transmission, full-modem loopback, and parallel port loopback testing. You *must* attach a 96-pin loopback plug to the card under test before running this test (See Appendix A, “Loopback Connectors”).
- ***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 section 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, “Loopback Connectors”). This test cannot be run concurrently with the Echo-TTY option enabled.

Note – TTY connection to the `spif` serial port requires corresponding character size set up. For example, if a TTY attachment is running with 8-bit character size, then the `spif` test option ‘Char Size’ should also be set to 8 bits.

- ***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. Type anything on your TTY keyboard, and the characters you type should show up on the TTY screen.

Note – If you do not type anything within two minutes, this test will time-out. If you do not type anything at all, this test will still eventually time-out.

This test is terminated 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 `spif` options menu.

You can also change the test options by modifying the `/opt/SUNWvts/bin/.customtest` file. See the section on Adding Your Own Tests in `.customtest` in the *SunVTS 2.0 User's Guide*.

spif *Test Modes*

Connectivity Mode

Not available for spif.

Online Mode

Not available for spif.

Offline Mode

The spif test supports only the Offline mode.

spif *Command Line Syntax*

/opt/SUNWvts/bin/spif *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*

Argument	Explanation	
dev=*device_name*	Specifies the serial ports in SBus card slots (0-63) being tested. Since there is no default, you must type a device name—either a board (stc0-7) or an individual port (term/0 - term/63):	
	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
		or
	/dev/term/*mm*	
	Where *mm* is 0 to 63 (any of the serial ports in SBus card slots)	

Argument	Explanation *(Continued)*
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 to 1 or 2; the default is 1.
Size=*character_size*	Sets character size as a number between 5 and 8.
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.

`spif` *Error Messages*

These error messages are generated when the SunVTS Serial Parallel Controller discovers errors. The error descriptions below identify probable causes for the card or test failure, and identify the Field Replaceable Unit (FRU), if possible. The three FRUs affected are: the Serial Parallel Controller card, the 96-pin shielded cable, and the Patch panel.

```
6000  <error_message>

6001  Expected = <value>, observed = <value>

6002  Expected (<value>):

6003  Observed (<value>):

6004  Modem Loopback test failed on <device_name>

6005  Off-line error on device <device_name>

6006  Paper out error on device <device_name>
```

```
6007  Busy error on device <device_name>

6008  Error on device <device_name>

6009  retry <number> exceeds limit <number> rc =s <value>

6010  Expected <number> bytes, observed <number> bytes

6011  Internal test failed on <device_name>

6012  Data Loopback test failed on <device_name>

6013  ioctl TIOCSSOFTCAR error on <device_name>

8000  Must provide device name!

8001  No SPC/S card found or device driver not installed

8002  ioctl TCSETS failed on <device_name>

8003  Open error on <device_name> - device driver may not
be installed properly

8004  Device <device_name> is already opened

8005  ioctl STC_SREGS(STC_IOREG-COR2) error in
<device_name>

8006  ioctl STC_GREGS(STC_IOREG-CCR) error in <device_name>

8007  ioctl STC_S1REGS(STC_IOREG-CCR) error in
<device_name>

8008  ioctl STC_SREGS(STC_PPCREG-PDATA) error in
<device_name>

8009  ioctl STC_GREGS(STC_PPCREG-PDATA) error in
<device_name>

8011  ioctl TIOCMGET failed on <device_name>
```

8012 <device_name>: Expected DSR set, observed clear

8013 <device_name>: Expected DSR clear, observed set

8014 ioctl TIOCMSET failed on <device_name>

8015 ioctl TIOCMBIC failed on <device_name>

8016 Read error on <device_name>

8017 Write error on <device_name>

8018 ioctl STC_GPPC error on device <device_name>

8019 Can't kill all processes

8020 Timeout error on <device_name>.

Probable_Cause(s):

a. No loopback plug

Recommended_Action(s):

a. Check Loopback plug!

8021 Timeout error on <device_name>.

Probable_Cause(s):

a. tty terminal not connected

Recommended_Action(s):

a. Check tty terminal connected to.

8022 Timeout error on <device_name>

sptest 24

*Serial Ports Test (*sptest*)*

This test checks the system's on-board serial ports (zs0, zs1), as well as any Multi-Terminal Interface (ALM2) boards (mcp0, mcp1, mcp2, mcp3). It writes data to the source device and then reads it back from the receiver device, verifying the data after each byte sent.

Intervention Mode must be enabled before clicking the option window button because sptest is not a scalable test.

Syncloop Testing

With this release of SunVTS, sptest has been enhanced to support syncloop testing. Syncloop testing involves sending and receiving data packets as HDLC frames over synchronous serial lines. The syncloop test to sptest runs in three phases.

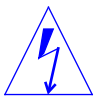
- The first phase listens to the port for any activity. If no activity is seen for at least four seconds, the test proceeds to the next phase. Otherwise, you are informed that the line is active and the test cannot proceed. sptest exits with an error if the line is active.
- The second phase is called the first packet phase, in which the test attempts to send and receive one packet. If no packets are seen after five attempts, the test exits with an error message. 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 issues an error message.

- The third phase is the multiple packet phase, where the test attempts to send many packets through the loop. Because the test has verified the integrity of the link in the first packet phase, the test will not fail after a particular number of time-outs. If a packet is not seen after the wait time expires, a message is displayed. The number and size of the packets sent during this phase is determined by a default value. Each packet is compared with its original for length and content. If a mismatch is detected, the test issues an error message.

Syncloop Testing Software Requirements

Syncloop testing requires that the `zsh0` and `zsh1` devices are located in `/dev`. If you don't have these devices configured, make sure the following two lines are in the `/etc/devlink.tab` file:

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



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

When these lines have been added to the `/etc/devlink.tab` file:

- **Change directories to** `/kernel/drv`, **and execute the** `add_drv zsh` **command.**

If this command does not work at first,

- **Execute the** `rem_drv zsh` **command and then execute the** `add_drv zsh` **command again.**

`sptest` *Options*

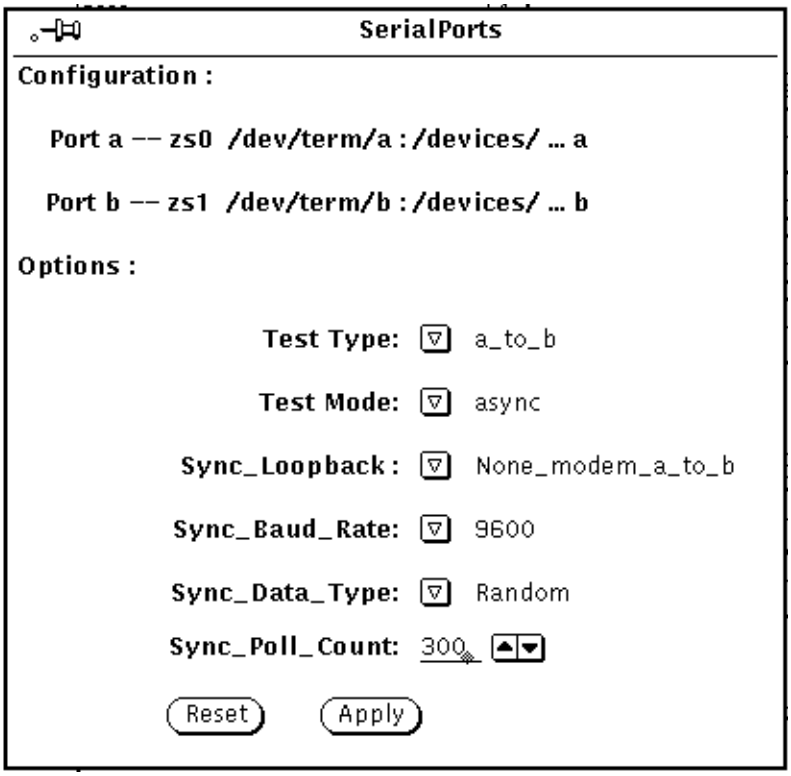


Figure 24-1 `sptest` Option Menu

The loopback settings refer to serial port testing. You must use loopback connectors (described in Appendix A, “Loopback Connectors“) to connect the CPU board ports you choose. The default is to link ports A and B with a loopback cable, as indicated by a-b. If you choose a only or b only, the SunVTS exerciser tests just that port, and expects a loopback connector on that port. If you choose a b, the program tests ports A and B separately, and expects a loopback connector on each port.

Loopback Assignments

In the following table, *a* represents port a of the CPU board, *b* represents port b of the CPU board, and the device names of the ports for each CPU board are listed.

Table 24-1 Loopback Settings

CPU	Port	Async Device	Sync Device
0	a b	zs0 zs1	zsh0 zsh1
1	a b	zs2 zs3	zsh2 * zsh3 *
2	a b	zs4 zs5	zsh4 * zsh5 *
* Currently, only zsh0 and zsh1 are supported by device drivers.			

- ***Test Type*** selections are a, b, a_b (a and b), a_to_b (a to b). An a to b cable is required if you select a_to_b.
- ***Test Mode*** is synchronous testing, asynchronous testing, or both.
- ***Sync Loopback***
  - Internal test — works with Test Mode a, b, a_b
  - Plug_a_to_a__b_to_b — a, b, a_b, test types require a loopback plug
  - no_modem_a_to_b — testing a_to_b and b_to_a requires a sync a to b cable; no modem is required
- ***Loopback Plug*** when enabled, provides data transmission loopback testing of the selected serial port. You must attach a loopback connector (See Appendix C) to the serial port being tested. This option is available only in synchronous mode.
- ***Sync Internal Test*** when enabled performs a quick internal check of the serial ports on the CPU board. You do not need to connect any loopback connectors to the serial ports to perform this test. This option is available only in synchronous mode.
- ***Sync a to b Modem Loopback*** is enabled if the Transmit and Receive clock sources are external (via modem). This option requires that one of the local modems or the remote modems be set in a loopback configuration. This option is available only in synchronous mode.

- ***Sync Baud Rate*** sets the baud rate from a menu. The choices are: 110, 300, 600, 1200, 2400, 4800, or 9600 baud. If you have changed the baud rate, the new value is used. This new value may not correspond to what the Baud Rate menu displays.

Note – You can only change the baud rate if you are testing in synchronous mode. If you are testing in asynchronous mode, the `sptest` uses the default baud rate (usually 9600).

- ***Sync Data Type*** specifies the selected hexadecimal type pattern. The choices are 0x55555555, 0xaaaaaaaa, or random. 0x55555555 and 0xaaaaaaaa are abbreviated on the option menu. This option is only available in synchronous mode.
- ***Sync Poll Count*** sets a value to determine how long the test will poll before reporting an error. Increase the value when running a heavy system load; reduce the value when running a light system load.

`sptest` *Test Modes*

Sptest supports all these three modes. It uses different test strategy on the serial port according to different mode. Following is the description of sptest under each mode:

Connectivity Mode

In this mode, the `sptest` tries to open the port to determine whether or not the device is connected. If the attempted open failed, a message is returned indicating that the port is not connected. If the port either opens or fails with a busy or exclusive use error, then the port is considered connected and the test mode returns a passed message.

Online Mode

In Online mode, `sptest` tries to minimize the impact on the application being tested. The Online test is separated by `sptest` into synchronous and asynchronous tests. In synchronous test mode, `sptest` opens the port and does the same test as the Offline test. In asynchronous test mode, `sptest` opens the port and checks for exclusive use of the port.

Offline Mode

In Offline mode, `sptest` performs the regular test, such as checking port activity, sending the first packet, and then sending multiple packets.

`sptest` *Command Line Syntax*

`/opt/SUNWvts/bin/sptest` *standard_arguments* **`-o dev=`***device_name*, **`porta=`***port_name*`,T=a,b,a_to_b,a_b,` **`L=`***test_loopback*`,` **`M=`***mode*`,` **`B=`***baud_rate*`,` **`I=`***loopback_pattern*

Argument	Explanation

`dev=`*device_name*	Specifies which serial port(s) to test. There is no default value; you must specify a device name such as:
	`zs0, zs1`
	`zs2, zs3`
	`..., ...`

`porta=`*port_name*	Specifies the name of port a of this CPU, i.e. term/a, term/b
`M=`*mode*	The default test mode is asynchronous. Specify one of the following modes: asynch synch both
`T=`*test_type*	**`Specifies the type of test, by number:`** a b a-to-b a-b
`L=`*test_loopback*	None Internal Plug Modem

Argument	Explanation *(Continued)*
B=*baud_rate*	Specifies one of the following baud rates for synchronous mode: **110** **300** **600** **1200** **2400** **4800** **9600** **19200** **38400** **Note:** If you are testing in asynchronous mode, the default baud rate is used (usually **9600**).
I=*loopback_pattern*	The default test data pattern is **r**. You can specify the following: **5** = 0x55555555 **a** = 0xaaaaaaaa **r** = random

`sptest` *Error Messages*

```
6000 Poll device <device_name> error <error_message>

6001 device <device_name> has error

6002 data error on device '<device_name>', exp = <value>,
obs = <value>.

6003 data miscompare, cnt = <number> exp = <value>, obs =
<value>
```

Probable_Cause(s):

a. **Bad serial port**

b. **System software**

```
6004 can't get sync mode info for <device_name>
```

```
6005 can't set sync mode info for <device_name>

6006 packet write failed, errno <number>

6007 len <value> should be <value>

6008 Received:

6009 Expected:

6010 <number> packets with wrong data received!

6011 <number> packets left in outbound queueing

6012 <number> packets lost in inbound queueing
```

Probable_Cause(s):

a. **System load too heavy**

b. **Cable connection error**

c. **Board error**

Recommended_Action(s):

a. **Rreduce system load.**

b. **Ccheck cable connection**

c. **If the problem persists, call your authorized Sun service provider.**

```
6013 Device '<device_name>' not responding.  No data
detected after <number> seconds.  Received <number> bytes
out of <number> sent.
```

Probable_Cause(s):

a. **System load too heavy**

b. **Loopback plug missing or defective**

c. **Bboard error**

d. **Bad serial port**

e. **System software**

Recommended_Action(s):

a. **Reduce system load**

b. **Check loopback plug**

c. **If the problem persists, call your authorized Sun service provider.**

```
8000 cannot open <device_name>, <error_message>

8001 Test Type a_to_b is not allowed on Quick Test!

8002 Couldn't open file '<device_name>': <error_message>

8003 Transmit failed on '<device_name>'.
```

24

sunbuttons 25

*SunButtons Test (*sunbuttons*)*

This test verifies that the SunButtons™ graphics manipulation device is working correctly.

From a .customtest *File*

In the non-interactive mode (from a .customtest file) this test verifies that each button functions; each button lights up in a round-robin fashion.

Here is an example of a .customtest entry for sunbuttons:

```
sunbuttons;sunbuttons;.text<TEXT|20|-s|>
```

From a Shell Command Line

This is an interactive test. The test displays a screen representation of the buttonbox (see Figure 25-1) where you can press each of the buttons and see the corresponding button's display change. 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.

To run the test, select the Diagnostics button. As the test runs, each button lights in a round-robin fashion.

Be sure that the buttonbox is connected to one of the serial ports, and that the buttonbox has a power transformer.

There is no option menu for this test.

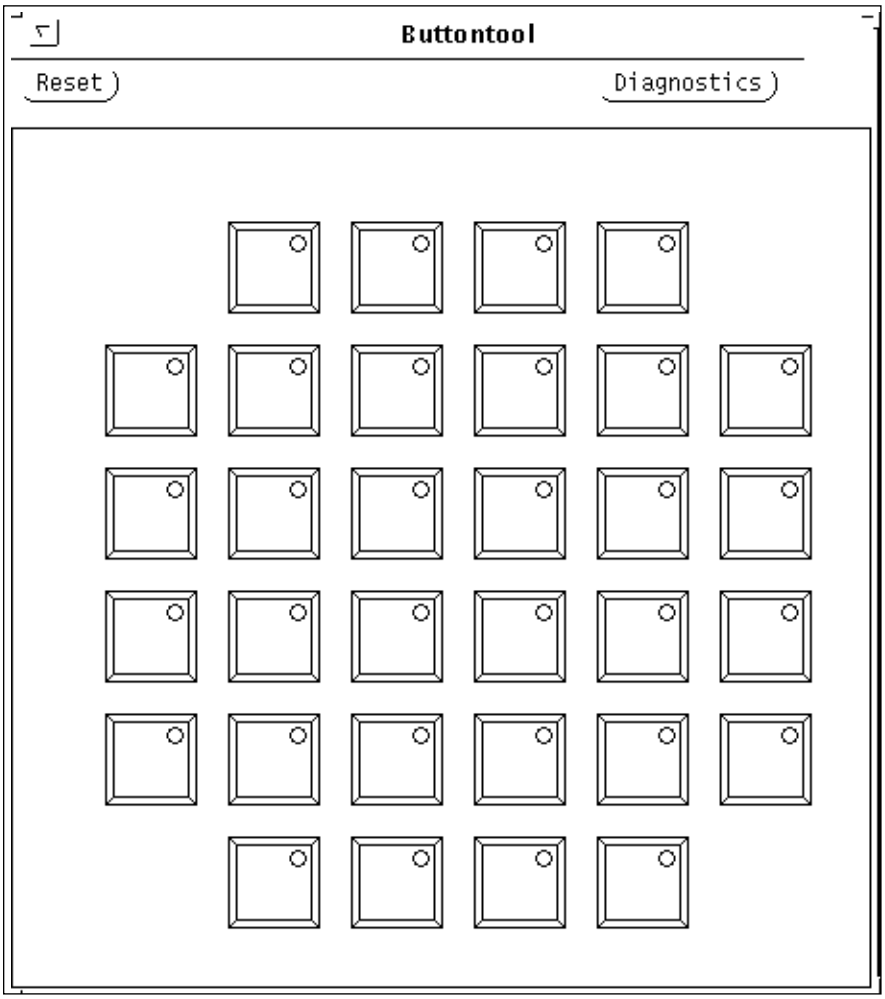


Figure 25-1 `sunbuttons` Test Menu

sunbuttons *Command Line Syntax*

/opt/SUNWvts/bin/sunbuttons diag *standard_arguments*

sunbuttons *Test Modes*

Connectivity Mode

Not available for sunbuttons.

Online Mode

Not available for sunbuttons.

Offline Mode

In the offline mode, the sunbuttons test verifies the functionality of each button, and that each button lights up in a round-robin fashion.

sunbuttons *Error Messages*

6000 Cannot open device.

6001 ioctl(VUIDSFORMAT, VUID_NATIVE)

6002 Writing Diag Command Failed.

6003 No Response from Buttonbox.

6004 Selftest failed.

6005 Weird unknown kind of error.

6006 Can not open device /dev/db.

6007 ioctl(VUIDSFORMAT, VUID_NATIVE) -- you may need to run bdconfig.

25

sundials 26

*SunDials Test (*sundials*)*

From a .customtest *File*

This test verifies that the SunDials™ graphics manipulation device controls are working properly. sundials also verifies the connection between the dialbox and serial port.

Here is an example of a .customtest entry for sundials:

```
sundials;sundials;text<TEXT|20|-s|>
```

From a Command Line

Running sundials from a command line starts an interactive test that displays a screen representation of the dialbox (see Figure 26-1). You can move each of the dials and see the corresponding dial's display change. To run the interactive test, select the Diagnostics Button on the top of the window representation. The dialbox dials on the screen will not move while the diagnostics test is running

Be certain that the dialbox is connected to one of the serial ports, and that the dialbox has a power transformer.

There is no option menu for this test.

The `sundials` test menu looks like this:

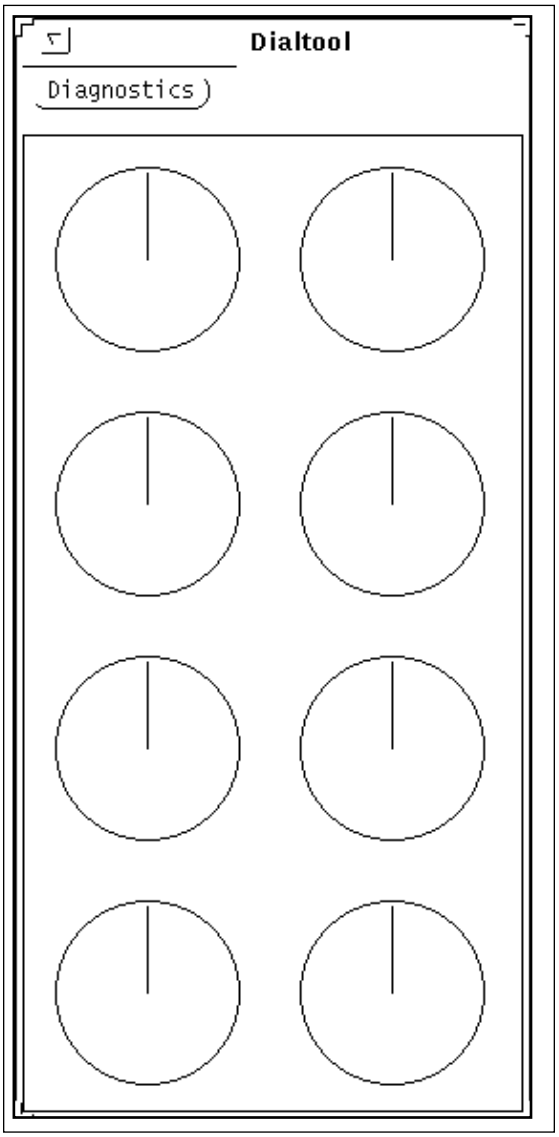


Figure 26-1 `sundials` Test Menu

sundials *Test Modes*

Connectivity Mode

Not applicable for this test.

Online Mode

Not applicable for this test.

Offline Mode

The sundials test verifies the connection between dialbox and serial port. The test also verifies all manipulation device controls.

sundials *Command Line Syntax*

/opt/SUNWvts/bin/sundials diag *standard_arguments*

sundials *Error Messages*

```
6000 Open failed for /dev/bd.

6001 ioctl(VUIDSFORMAT, VUID_NATIVE) error.

6002 ioctl(VUIDSFORMAT, VUID_EVENT) error.

6003 Could not talk to Dialbox.

6004 Cannot open device.

6005 Writing Diag Command failed.

6006 No response from Dialbox.

6007 Selftest failed.

6008 Weird unknown kind of error.
```

```
6009 ioctl(VUIDSFORMAT, VUID_NATIVE) -- you may need to run
bdconfig.
```

sunlink 27

*HSI/S Boards Test (*sunlink*)*

This test verifies the functionality of the SBus HSI boards. `sunlink` tests the HDCL and SDLC protocol of SBus HSI boards. `sunlink` downloads the DCP microcode, initializes the selected channel, and configures the selected channel to the protocol being tested.

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; other patterns may be specified. 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 8 minutes per port. `sunlink` does 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.

Note – 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 option menu must have loopback connectors attached. See Appendix A for loopback connector part numbers and wiring instructions.

`sunlink` *Configurations*

The Configurations field displays the available ports.

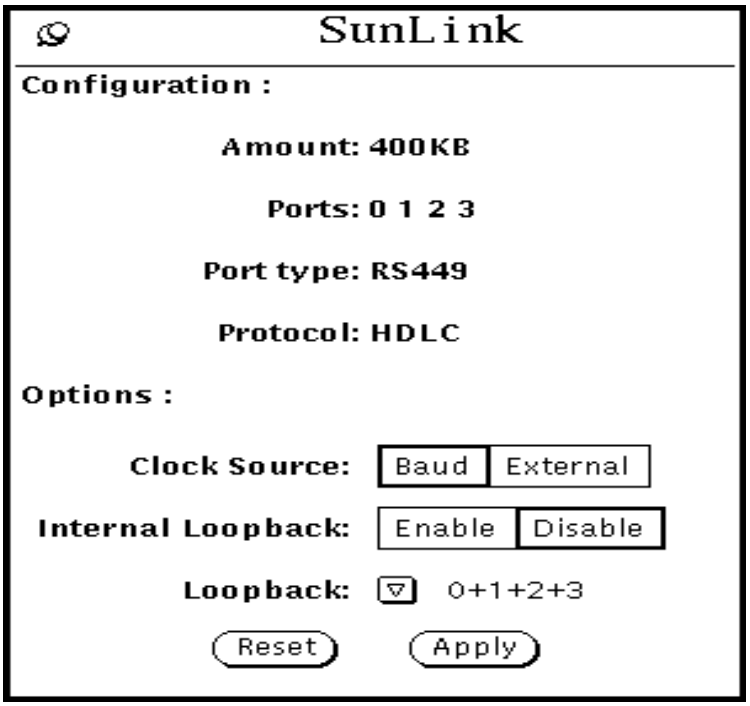


Figure 27-1 `sunlink` Option menu

- ***Clock Source*** lets you select either the on-board 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 on-board.
- ***Loopback*** specifies the loopback type: simple single external port loopback, multiple external port loopback, and port-to-port external loopback.

`sunlink` *Command Line Syntax*

`/opt/SUNWvts/bin/sunlink` *standard_arguments* **`-o`**
`dev=`*device_name*`,P=`*port#*`, p=`*data_pattern*`,I, C=`*B/E*

Argument	Explanation
`dev=`*device_name*	Specifies the device to be tested.
	`hih0` HDLC and SDLC protocols
`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/S (external).
`c=`*clocksource*	Specifies the clock source value as one of the following:
	B On-board clock source
	E External clock source

The following is a typical command line syntax for testing an HSI/S Sbus card:

```
# /opt/SUNWvts/bin/sunlink -o dev+hih0,P=0_to_1+2_to_3
```

Typing this at the command line tests the internal loopback for port 0.

27

sunlink *Test Modes*

Connectivity Mode

Not available for sunlink.

Online Mode

Not available for sunlink.

Offline Mode

The sunlink test only supports the Offline mode.

sunlink *Loopback Connectors*

Refer to Appendix A of this manual, and the High Speed Serial Interface hardware manuals for information on loopback connectors.

sunlink *Error Messages*

```
6000  Device name must be provided!

6001  Invalid port specification.

6002  Invalid port number.

6003  BSC protocol is not implemented for <device_name>.

6004  ASYNC protocol is not implemented for <device_name>.

6005  Illegal protocol specified for ‘<device_name>’.

6006  Could not open file : <file_name>

6007  Could not open file devname <device_name>

6008  Could not open file (clonename)<file_name>

6009  Can’t get sync mode info for <device_name>.
```

6010 Can't set sync mode info for <device_name>.

6011 Couldn't successfully execute '/usr/sunlink/dcp/dcp<file_name>.

6012 Packet received but none sent! Activity on-line. Quiesce other end before starting.

6013 Transmit failed on '<device_name>'.

6014 Receive failed on '<device_name>'.

6015 Data compare error on '<device_name>', exp = <value>, actual = <value>, offset = <number>.

6016 <device_name> does not respond

Probable_Cause(s):

a. No loopback plug

b. System load too heavy

Recommended_Action(s):

a. Check Loopback plug

b. Reduce system load

6017 sigalrm: TIMEOUT

Probable_Cause(s):

a. No loopback plug

b. System load too heavy

Recommended_Action(s):

a. Check Loopback plug

b. Reduce system load

8000 Unsupported primitive type <value>

```
8001   <text> (<error_message>)

8002   Response ctl.len too short:   <value>

8003   DL_INFO_ACK was not M_PCPROTO

8004   DL_OK_ACK was not M_PCPROTO
```

*Pixel Processor Test (*sxtest*)*

This test checks models of SPARCstation 10 and SPARCstation 20 machines equipped with an onboard Pixel Processor module. `sxtest` is specific to the VSIMM (Video SIMM)/SX Memory Controller) devices in the SPARCstation 10 SX and SPARCstation 20 SX.

Note – Because of possible conflicts between `cg14` SunVTS frame buffer tests and OpenWindows applications that use the `cg14` frame buffer, the following restrictions apply when running the `sxtest` SunVTS test:

Do not run any graphic applications other than OpenWindows while running SunVTS frame buffer tests.

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

Do not run any OpenWindows 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.

Enable the frame buffer locking option from the Options window for the system console `cg14` device (see “FB Locking” on page 335).

If `sxtest` is run with VRAM enabled, then frame buffer locking *must* also be enabled.

This test locates load error, store error, ALU error, logic error, and so on, of the Pixel Processor by reading and verifying data from the control registers of the Pixel Processor, virtual memories, or video memories. This test also verifies the integration function of the cg14 frame buffer and its device driver, video memories, and data memories. `sxtest` also writes a test pattern to the frame buffer for visual verification. The accuracy of this test is dependant on other hardware, such as the CPU, memory card(s), and `cg14` graphics card. Run `cg14test` and other tests before running `sxtest`. `sxtest` is a series of 13 modules, described below.

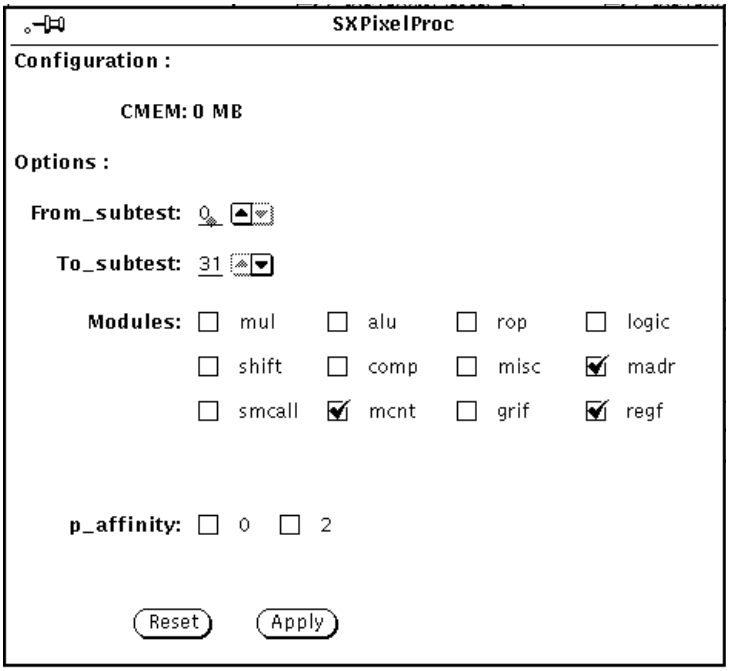


Figure 28-1 `sxtest` Option Menu (CMEM:0MB, none, `cg14` board)

Note – The `sxtest` option menu is customized, based on the configuration of the test system.

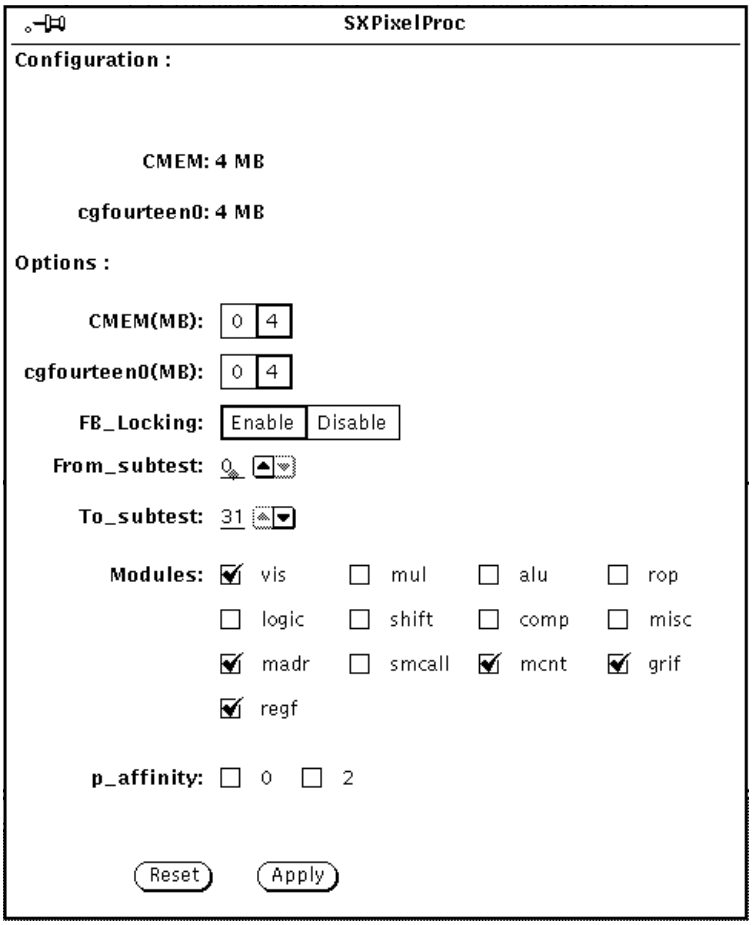


Figure 28-2 `sxtest` Option Menu (CMEM:4MB, cg14:4MB)

FB Locking

See the section about testing multiple frame buffers in *SunVTS 2.0 User's Guide* for details.

Frame buffer locking is enabled by default on the window server running the OpenWindows software.

CMEM (Contiguous Memory)

Choose either 0 or 4, if your system has 4 or more Mbytes of contiguous memory available.

To set CMEM, enter the following:

```
/usr/platform/SUNW,SPARCstation10,SX/sbin/sxconfig
```

The system must have 32 Mbytes of memory or more to set CMEM. For example:

```
% sxconfig -s 4 -1 28 -f
```

VRAM (Video Random Access Memory)

Choose either 0|4|8 video random access memory.

Caution – If `sxtest` is run with its VRAM enabled, then frame buffer locking *must* be enabled or SunVTS results in errors.

`sxtest` *Module Descriptions*

The 13 `sxtest` modules are described in this section.

Display (Module0)

♦ **Click enable to display visual patterns.**

Three subtests call the SPAM library and display pictures to verify the integrity of a subset of the kernel and the SPAM libraries via the SPARCstation 10 SX video system. These routines are ported from the SPAM demonstration programs. All subtests in this module are skipped if the cg14 frame buffer does not exist, or if the VRAM is disabled.

One of the following subtests displays pictures between other subtests. This helps you determine whether the system is hung or if `sxtest` is running.

- `rect_test` — The screen is filled with random rectangles. The rectangles are drawn in CHUNKY_XBGR mode if 32-bit mode OpenWindows is running. If not, they are drawn in CHUNKY_C8 mode with the SPAM library routine `sl_rect_fill_32`.
- `shaa` — A picture of shaded lines is drawn in CHUNKY_BGR mode with the SPAM library routines `sl_line_shaa_32`, `sl_span_load_8` and `sl_rect_fill_8`.

Note – The `shaa` test is skipped if the test is running on an 8-bit window.

- `lines` — The screen is filled with lines of various colors. These lines are drawn in CHUNKY_XBGR mode if 32-bit mode OpenWindows is running; If not, they are drawn in CHUNKY_C8 mode with SPAM library routine `sl_line_fill_8`.

MUL (Module1)

♦ **Click enable to test the multiplier operations.**

Eight subtests are called, and each subtest has 2500 randomly generated MUL SPAM macros.

- `sp_mul0`
- `sp_mul1`
- `sp_mul2`
- `sp_mul3`
- `sp_mul4`
- `sp_mul5`
- `sp_mul6`
- `sp_mul7`

Each subtest tests SPAM MUL instruction sets by executing random SPAM MUL macro patterns, for example:

```
spam_dot(S_0,R42,R45,R31,5)
spam_mulr(L_16,R44,R29,R52,1)
spam_mul(S_15,R115,R114,R58,4)
spam_mul(L_16,R89,R110,R81,8)
spam_mulr(S_8,R21,R76,R53,1)
spam_saxpr(S_8,R54,R46,R98,2)
spam_dotr(L_16,R75,R40,R20,5)
```

```
spam_dot(L_16,R44,R45,R84,4)
spam_saxp(L_0,R93,R96,R44,8)
spam_mulr(L_0,R86,R56,R56,5)
spam_dotr(L_0,R14,R62,R40,2
spam_saxpr(S_15,R112,R85,R95,7)
```

ALU (Module2)

♦ **Click enable to test ALU operations.**

Five subtests are called, and each subtest has 2500 randomly generated ALU SPAM macros.

- `sp_alu0`
- `sp_alu1`
- `sp_alu2`
- `sp_alu3`
- `sp_alu4`

Each subtest tests SPAM ALU instruction sets by executing random SPAM ALU macro patterns, for example:

```
spam_subv(R101,R31,R42,1)
spam_subs(R90,R44,R90,14)
spam_subv(R44,R70,R29,14)
spam_sum(R58,R95,R114,9)
spam_adds(R54,R46,R98,10)
spam_addi(R9,51,R68,9)
spam_abs(R76,R28,7)
spam_addv(R80,R59,R93,11)
```

ROP (Module3)

♦ **Click enable to test the ROP operations.**

Five subtests are called, and each subtest has 2500 randomly generated ROP SPAM macros.

- `sp_rop0`
- `sp_rop1`
- `sp_rop2`
- `sp_rop3`

- sp_rop4

Each subtest tests SPAM ROP instruction sets by executing random SPAM ROP macro patterns, for example:

```
spam_selb(R101,R31,R42,1)
spam_ropl(R90,R27,R44,14)
spam_sels(R19,R16,R112,15)
spam_ropm(R47,R29,R96,16)
spam_selb(R52,R43,R29,5)
spam_ropb(R115,R114,R58,7)
spam_selv(R57,R75,R16,2)
spam_ropm(R110,R93,R83,13)
```

LOGIC (Module4)

♦ **Click enable to test the logical operations.**

Five subtests are called, and each subtest has 2500 randomly generated LOGIC SPAM macros.

- sp_logic0
- sp_logic1
- sp_logic2
- sp_logic3
- sp_logic4

Each subtest tests SPAM LOGIC instruction sets by executing random SPAM LOGIC macro patterns, for example:

```
spam_xors(R101,R31,R42,1)
spam_xori(R90,101,R90,14)
spam_xorv(R30,R19,R95,13)
spam_ands(R108,R16,R125,1)
spam_andv(R115,R114,R58,7)
spam_ors(R46,R89,R8,16)
spam_orv(R57,R75,R16,2)
spam_andi(R9,51,R68,9)
```

SHIFT (Module5)

♦ **Click enable to test the shift operations.**

Five subtests are called, and each subtest has 2500 randomly generated SHIFT SPAM macros.

- `sp_shift0`
- `sp_shift1`
- `sp_shift2`
- `sp_shift3`
- `sp_shift4`

Each subtest checks SPAM SHIFT instruction sets by executing random SPAM SHIFT macro patterns, for example:

```
spam_sllv(R101,R31,R42,1)
spam_slli(R90,5,R90,14)
spam_srai(R30,19,R95,13)
spam_srli(R108,16,R125,1)
spam_sllv(R52,R43,R29,5)
spam_slfi(R46,25,R8,16)
spam_slfs(R57,R75,R16,2)
spam_srav(R54,R44,R93,8)
spam_srlv(R58,R60,R96,16)
```

COMP (Module6)

♦ **Click enable to test the compare operations.**

Five subtests are called, and each subtest has 2500 randomly generated COMP SPAM macros.

- `sp_comp0`
- `sp_comp1`
- `sp_comp2`
- `sp_comp3`
- `sp_comp4`

Each subtest checks SPAM COMP instruction sets by executing random SPAM COMP macro patterns. For example:

```
spam_cmpv_gt(R101,R31,R42,1)
spam_cmps_lt(R90,R44,R90,14)
spam_cmps_eq(R95,R112,R19,12)
spam_cmpv_gt(R44,R43,R29,14)
spam_cmpv_lt(R115,R114,R58,7)
spam_cmps_gt(R46,R89,R8,16)
spam_cmps_eq(R57,R75,R16,2)
spam_cmpv_le(R54,R46,R98,10)
spam_cmpv_eq(R9,R51,R68,9)
spam_cmps_gt(R76,R103,R28,7)
spam_cmpv_eq(R52,R37,R50,8)
spam_cmpv_ge(R61,R86,R16,12)
```

MISC (Module7)

♦ **Click enable to test the miscellaneous operations.**

Five subtests are called, and each subtest has 2500 randomly generated MISC SPAM macros.

- `sp_misc0`
- `sp_misc1`
- `sp_misc2`
- `sp_misc3`
- `sp_misc4`

Each subtest checks SPAM MISC instruction sets by executing random SPAM MISC macro patterns, for example:

```
spam_scat(R45,-1,R29,1)
spam_gath(R95,-6,R114,9)
spam_delt(R89,R9,R16,16)
spam_plot(R54,R46,R98,10)
spam_plot(R53,R20,R75,16)
spam_scat(R91,-2,R70,9)
spam_gath(R120,-2,R51,15)
spam_delt(R59,R95,R120,1)
```

MADR (Module8)

♦ **Click enable to test the address lines of** `sx`**.**

Eight subtests are called; each subtest verifies 0x100000 SPAM address with `spam_stld` and `spam_ldld` instructions. All address bits and data bits of 4 Mbytes of VRAM and 4 Mbytes of DRAM are tested after running through the eight subtests.

- `0x00000000-0x000fffff`
- `0x00100000-0x001fffff`
- `0x00200000-0x002fffff`
- `0x00300000-0x003fffff`
- `0xfc000000-0xfc0fffff *`
- `0xfc100000-0xfc1fffff *`
- `0xfc200000-0xfc2fffff *`
- `0xfc300000-0xfc3fffff *`

* These subtests are skipped if the CMEM option is disabled.

SMCALL (Module9)

♦ **Click enable for a brief test of `sxtest` functionality.**

Eleven subtests are called from `spam.smcall` to verify the general function of the SMC chip. All subtests have a cg14 version and a non-cg14 version. These subtests repeat four times, each time with the `IQ FIFO` programmed to a different number of entries (8, 16, 32, or 64).

- `shift_ldst`
- `instr_mix *`
- `arith_ldst`
- `cmp_ldst`
- `select_ldst`
- `interlock_all *`
- `logic_ldst`
- `mult_ldst`
- `rop`
- `scat_ldst`
- `delt_ldst`

* These subtests are skipped if the VRAM option is set to disable.

MCNT (Module10)

♦ **Click enable to test the load and store functions with different repeat counts.**

Twelve subtests are called; they test the SPAM store functions by varying address offset and item count.

- `spsd_stba_cnt`
- `spsd_stbd_cnt`
- `spsd_stbds_cnt`
- `spsd_stcd_cnt`
- `spsd_stla_cnt`
- `spsd_stld_cnt`
- `spsd_stlds_cnt`
- `spsd_stpd_cnt`
- `spsd_stqd_cnt`
- `spsd_stsa_cnt`
- `spsd_stsd_cnt`
- `spsd_stsds_cnt`

GRIF (Module11)

♦ **Click enable to test the graphic interface logic.**

Thirty-six subtests are called; they test the SPAM graphic interface login with load/store instructions. All subtests are skipped if `cg14` doesn't exist.

- `spsd_stbd_dram`
- `spsd_stbd_xbgr`
- `spsd_stbd_bgr`
- `spsd_stbd_8x`
- `spsd_stbd_8c`
- `spsd_stbd_x32`
- `spsd_stbd_b32`
- `spsd_stbd_g32`
- `spsd_stbd_r32`
- `spsd_stsd_dram`
- `spsd_stsd_xbgr`
- `spsd_stsd_bgr`
- `spsd_stsd_8x`
- `spsd_stsd_8c`
- `spsd_stsd_x32`
- `spsd_stsd_b32`
- `spsd_stsd_g32`
- `spsd_stsd_r32`
- `spsd_ldbd_dram`
- `spsd_ldbd_xbgr`
- `spsd_ldbd_bgr`
- `spsd_ldbd_8x`
- `spsd_ldbd_8c`
- `spsd_ldbd_x32`
- `spsd_ldbd_b32`
- `spsd_ldbd_g32`
- `spsd_ldbd_r32`
- `spsd_ldsd_dram`
- `spsd_ldsd_xbgr`
- `spsd_ldsd_bgr`
- `spsd_ldsd_8x`
- `spsd_ldsd_8c`
- `spsd_ldsd_x32`
- `spsd_ldsd_b32`

- spsd_ldsd_g32
- spsd_ldsd_r32

REGF (Module12)

♦ **Click enable to test the register file pointer logic.**

Twenty-two subtests are called from `spam.regfile` to verify the register file's logic with assorted SPAM instructions.

- readpointer1 *
- readpointer2 *
- readpointer3 *
- readpointer4 *
- writepointer1 *
- writepointer2 *
- writepointer3 *
- writepointer4 *
- readpointer5 *
- writepointer5 *
- rdptr0 †
- wrptr0 †
- rdptr1 †
- wrptr1 †
- rdptr2 †
- wrptr2 †
- rdptr3 †
- wrptr3 †
- rdptr4 †
- wrptr4 †
- rdptr5 †
- wrptr5 †

* These subtests are skipped if the VRAM option is disabled.

† These subtests are skipped if the CMEM option is disabled.

`sxtest` *Test Modes*

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

Connectivity Mode

Not available for `sxtest`.

Online Mode

Not available for `sxtest`.

Offline Mode

In this mode, `sxtest` runs the visual subtest, memory address subtest, memory count subtest, graphic interface subtest, and register file subtest by default. The user can select or de-select other subtests as needed.

28

sxtest *Command Line Syntax*

/opt/SUNWvts/bin/sxtest *standard_arguments* **-o dev=***device_name*, **lock=***E(nable)/D(isable)*, **tm=***to_module#*, **cmem=***n*, **vram**=*n*, **md**=*XXX*

Argument	Explanation
dev=*device_name*	Specifies the device_name to be tested.
lock=*E(nable)/D(isable)*	Enables / disables frame buffer locking. See the section "Testing Multiple Frame Buffers" in Chapter 3 of the *SunVTS 2.0 User's Guide* for details. Frame buffer locking is enabled by default on the window server running the OpenWindows software.
tm=*to_module#*	Specifies an ending module number; use these last four arguments to narrow sxtest to a specific test scope.
cmem=*0 | 4*	Enables or disables the contiguous memory; choose either 1 to enable or 0 to disable. **Note:** You must choose disable (0) if your system is equipped with less then 4 Mbytes of contiguous memory.
cgfourteen0=*0 | 4 | 8*	Enables or disables the video random access memory; choose 1 to enable or 0 to disable.
md=*vis+alu+...*	Selects which modules are tested in a pass. XXX

sxtest *Error Messages*

```
6002 Unable to lock frame buffer.
```

Probable_cause(s):

a. FB lock was enabled while Window System was not running.

```
6004 Can`t get sem <number>

6006 Failed lock sem:dev <number>, type <number>

6008 Failed unlock sem

6010 Failed GETVAL sem=
```

```
6012 Failed sem_val

6014 console_p() failed :

6016 Failed sem_setall

6017 Failed sem_rmid

6018 Failed get sem

6020 ioctl command <name> failed

6022 <error_message>

6024 Failed to malloc for <name>

6026 mem ccitt[<name>+<number>,<name>+<number>]: o:<number>
e:<number> o^e:<number>

6028 mem crc16[<name>+<number>,<name>+<number>]: o:<number>
e:<number> o^e:<number>

6030 reg ccitt: o:<number> e:<number> o^e:<number>

6032 reg crc16: o:<number> e:<number> o^e:<number>

6034 Failed to open <name> device.

6035 First Open of sx device

6036 Modules selected improperly!!!

6038 <name>, i_module = <name>

6040 FB unlock request failed.

8000 Can't acquire console semaphore
```

Probable_cause(s):

a. Syetem load too heavy

b. System error

Recommended_Action(s):

a. Reduce the system load

b. If the problem persists, call your authorized Sun service provider.

```
8002 Can`t get sem <number>

8004 Unable to set page bounds

8006 mmap: unable to map MDI control register space

8008 Failed to mmap <name>

8010 sxtest is only running in offline mode.

8012 Failed to open <name>
```

systest

This test 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*

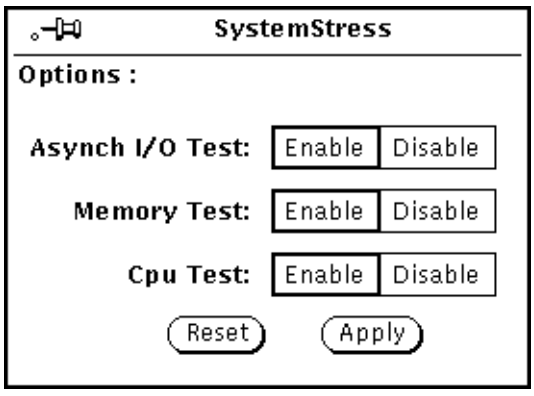


Figure 29-1 `systest` Option Menu

29

`systest` *Test Modes*

Connectivity Mode

Not available for `systest`.

Online Mode

Not available for `systest`.

Offline Mode

The Offline Mode is the only test mode available for `systest` .

`systest` *Command Line Syntax*

`/opt/SUNWvts/bin/systest` *standard_arguments*

`systest` *Error Messages*

```
6005 (<thread_id>)Couldn't open file /dev/mem:
<error_message>.

6006 (<thread_id>)'valloc' results incorrect:
<error_message>.

6007 (<thread_id>)Bad 'mmap' results: <error_message>, page
= <value>/<value>, addr = <address>.

6008 (<thread_id>)kvm_open() failed: <error_message>.

6009 (<thread_id>)kvm_nlist() failed: <error_message>.

6010 (<thread_id>)Defective namelist in '/vmunix'.

6011 (<thread_id>)Couldn't read physical memory list:
<error_message>.

6012 (<thread_id>)Couldn't read physical memory address.
```

6013 (<thread_id>)Couldn't read physical memory size: <error_message>.

6014 Cannot open <device_name>.

6015 Test Failed due to segment violation error.

Probable_Cause(s):

a. Lack of sufficient memory resources?

Recommended_Action(s):

a. Retry test after increasing available memory.

6016 Test Failed due to fpu bus error.

Probable_Cause(s):

a. Lack of sufficient memory resources?

Recommended_Action(s):

a. Retry test after increasing available memory.

6017 Test Failed due to fpu exception error.

Probable_Cause(s):

a. Lack of sufficient memory resources?

Recommended_Action(s):

a. Retry test after increasing available memory.

6018 Test Failed due to illegal instruction.

Probable_Cause(s):

a. Lack of sufficient memory resources?

Recommended_Action(s):

a. Retry test after increasing available memory.

29

```
8000 Failed system test (FPU, VMEM, I/O).
```

tapetest 30

*Tape Drive Test (*tapetest*)*

This test waits for 60 seconds to clear out any bus traffic, then it rewinds the tape, erases the tape, writes a pattern to a specified number of blocks (or, for a SCSI tape, writes to the end of the tape). Next, `tapetest` rewinds the tape and then reads and compares the information just written. The test expects a device name and number of sectors as input parameters. Next, `tapetest` writes to the device from a 126x512 byte buffer, then from a 512-byte buffer for any "leftovers." `tapetest` is not a scalable test.

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

`tapetest` *Options*

The SunVTS `tapetest` supports 4mm, 8 mm, 1/4" cartridge, and 1/2" front-load tape drive testing. The options available for each of the tape devices differ slightly. Examples of the option pop-up menus for some devices are shown below.

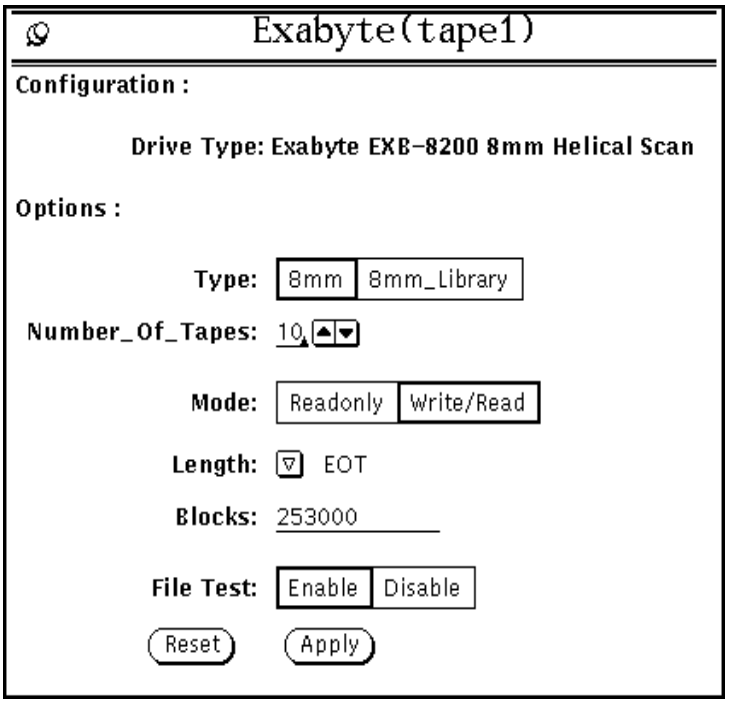


Figure 30-1 `tapetest` Option Menu (8mm tape drives)

Note – This test may take a while to stop after being disabled.

The example above shows the options menu for an 8mm tape drive. This menu differs from other tape drive option menus because it has no format or reconnect option choices.

If the drive is a half-inch front-load tape drive, the menu shown in Figure 30-2 will be displayed.

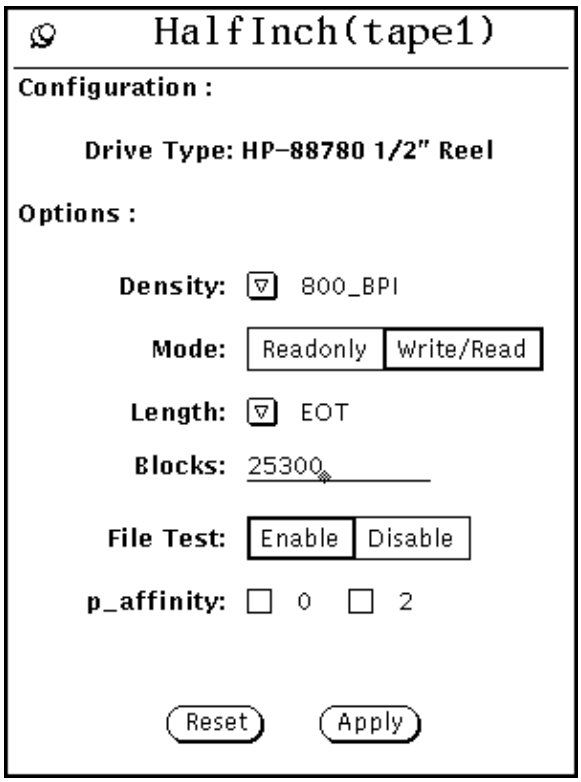


Figure 30-2 `tapetest` Option Menu (half-inch front-load tape drives)

A quarter-inch tape drive option menu looks like Figure 30-3.

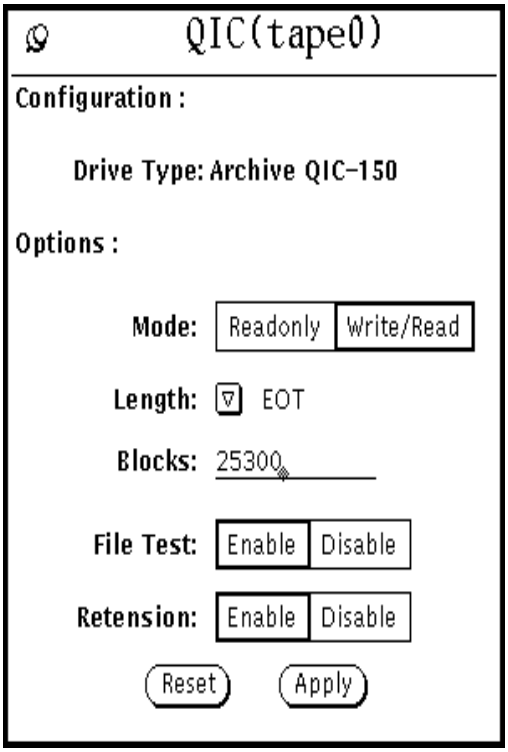


Figure 30-3 `tapetest` Option Menu (quarter-inch tape drives)

A 4mm tape drive option menu looks like Figure 30-4.

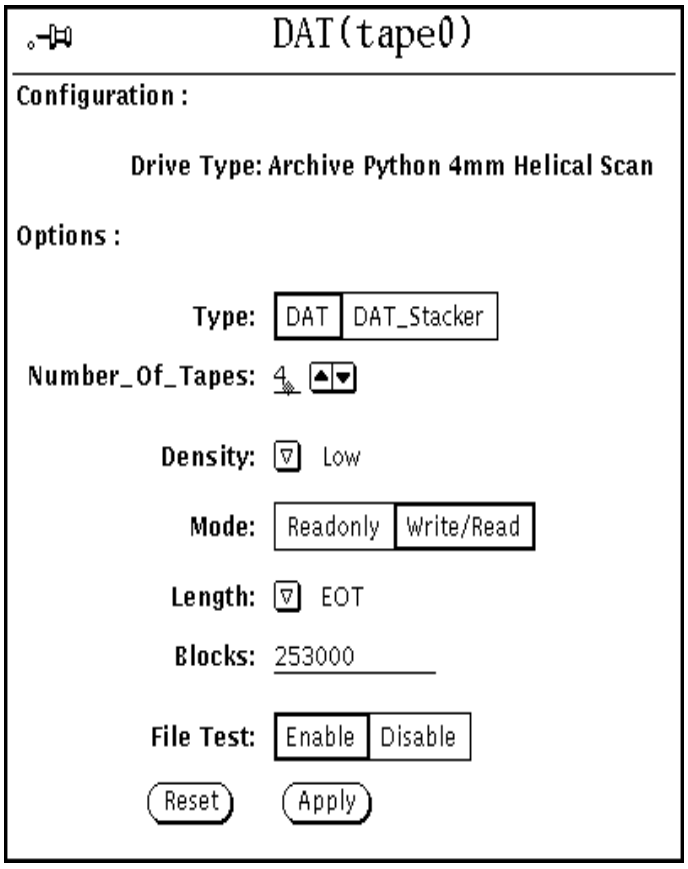


Figure 30-4 `tapetest` Option Menu (4mm tape drives)

Format:

QIC-11 and QIC-24 are quarter-inch tape formats that this test uses when it writes to the scratch tape you inserted. QIC-11 uses a 1-byte block ID; QIC-24 uses a 4-byte block ID, so each block on a QIC-24 tape is uniquely identifiable.

Use a standard scratch tape for this test.

QIC-11 format is the default testing format; however, you can choose QIC-24 only, or both of QIC-11 and QIC-24 formats. If you choose both, the test first writes one pass to the tape in QIC-11 format, and then writes a second pass over it in QIC-24 format.

Density

The following settings are available for 8mm tape drives:

`EXB8200`	Writes 2.3 Gbytes of data to the tape.
`EXB8500`	Increases the density and writes 5 Gbytes of date to the tape.
`Both`	Writes both 2.3 Gbytes, and 5 Gbytes of data to the tape.

For half-inch tape drives the available settings are `800`, `1600`, and `6250` BPIs.

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.

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.

File Test

The *tape file* test sequence is as follows:

- writes three files
- rewinds
- reads part of the first file

- forward spaces to the start of the second file
- reads the second file
- forward spaces to the start of the third file
- 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.

# of passes

If you have enabled the head cleaning option, you must enter the number of test passes the SunVTS exerciser should execute before suspending testing to provide time to clean the tape drive head.

`tapetest` *Test Modes*

The `tapetest` supports all three modes. It performs different test schemes on the network device, according to the mode you select.

Connectivity Mode

In this mode, `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.

Online Mode

In this mode, `tapetest` opens the device and reads several blocks from it. If the read operation is successful, the test has passed. On the other hand, if the device is busy or if no tape cartridge can be found in the drive, the test cannot be run and fails.

Offline Mode

In this mode, `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 does a read and compare of the pattern.

30

`tapetest` *Command Line Syntax*

`/opt/SUNWvts/bin/tapetest` *standard_arguments* **`-o dev=`***device_name*`,`**`s=`***block_count*`,`**`d=`***density*`,`**`m=`***mode*`,`**`l=`***length*`,`**`ft`**`=`*enables/disables*`,`**`ret`**`=`*enables/disables*`,`**`dat=`***dat_type*`,`**`8mm=`***8mm_type* **`num=`***magazine_size*

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)
`ft``=`*enables/disables*	Enables / Disables the file test
`ret``=`*enables/disables*	Enable / Disables tape retention
`dat=`*dat_type*	If you are testing a Digital Audio Tape drive, specify if it's a regular DAT drive or a DAT Stacker. The choices are **`DAT`** and **`DAT_Stacker`**.
`8mm=`*8mm_type*	If you are testing an 8mm tape drive, specify if it's a regular 8mm tape drive or a Tape Library. The command line choices are **`8mm`** and **`8mm_Library`**.
`num=`*magazine_size*	If you are testing a DAT Stacker, specify the magazine size.

`tapetest` *Error Messages*

`6000  Missing device name.`

Probable_Cause(s):

a. No device name specified.

Recommended_Action(s):

a. Retry Test with correct device name specification.

`6001  failed ioctl on unload and load next tape: err_code = <number>!`

Probable_Cause(s):

a. This Stacker/Library Unit may lack the Automatic Cycle Option.

Recommended_Action(s):

a. Run the test with Single Pass Option Enabled.

`6002  Cannot open <device_name>: <error_message>.`

Probable_Cause(s):

a. No tape media in drive.

Recommended_Action(s):

a. Load tape media and retry test.

`6005  Cannot close <device_name>: <error_message>.`

Probable_Cause(s):

a. Bad drive.

Recommended_Action(s):

a. Retry test, if problem persists check/replace the drive.

```
6006  <text> write failed on <device_name>, block <number>:
EOF reached.
```

Probable_Cause(s):

a. EOF reached prematurely.

Recommended_Action(s):

a. Retry Test, with reduced blocksize specification or until EOF.

```
6007  <text> write failed on <device_name>, block <number>:
<error_message>, sense key(0x<value>) = <text>.
```

Probable_Cause(s):

a. Drive needs attention.

Recommended_Action(s):

a. Perform action to reset sense key and retry Test.

```
6008  <text> read failed on <device_name>, block <number>:
EOF reached.
```

Probable_Cause(s):

a. EOF reached prematurely.

Recommended_Action(s):

a. Retry Test, with reduced blocksize specification or until EOF.

```
6009  <text> read failed on <device_name>, block <number>:
<error_message>, sense key(0x<value>) = <text>.
```

Probable_Cause(s):

a. Drive needs attention.

Recommended_Action(s):

a. Perform action to reset sense key and retry Test.

6010 <text> compare failed on <device_name>, block <number>, offset <number>, pattern 0x<value>, data= 0x<value>.

Probable_Cause(s):

a. Bad Drive

Recommended_Action(s):

a. Retry Test, if problem persists check/replace the drive.

6011 <device_name> tape MTIOCGETDRIVETYPE ioctl: <error_message>.

Probable_Cause(s):

a. Bad Drive

Recommended_Action(s):

a. Retry Test, if problem persists check/replace the drive.

6012 <text> failed on <device_name>: <error_message>.

Probable_Cause(s):

a. Bad Drive

Recommended_Action(s):

a. Retry Test, if problem persists check/replace the drive.

6013 Failed Connectivity test on <device_name>:<error_message>

Probable_Cause(s):

a. Broken Cable.

b. Bad Drive.

Recommended_Action(s):

a. Check/Replace Cable.

b. Check/Replace Drive.

```
6014  Failed Online test:<device_name>.
```

Probable_Cause(s):

a. Device may be in use.

Recommended_Action(s):

a. Retry Later.

```
6015  Failed Online test:<device_name>:<error_message>.
```

Probable_Cause(s):

a. No tape media in drive

Recommended_Action(s):

a. Load tape media and retry test.

```
6016  Online test was unsuccessful:<device_name>.
```

Probable_Cause(s):

a. No tape media in drive

Recommended_Action(s):

a. Load tape media and retry test.

```
6017  Failed Online read test:<device_name>.
```

Probable_Cause(s):

a. Bad Drive

Recommended_Action(s):

a. Retry Test, if problem persists check/replace the drive.

```
6018  Open Failed on <device_name>: <error_message>.
```

Probable_Cause(s):

a. Broken Cable.

b. Bad Drive.

Recommended_Action(s):

a. Check/Replace Cable.

b. Check/Replace Drive.

```
6018   recon : No Corresponding SCSI disk device.

6019  Needs both SCSI disk and SCSI tape to run the test.

6020   Couldn't retension '<device_name>'.

6021   Couldn't read file '<device_name>'.

6022   SCSI disconnect/reconnect failed.

6023   Spurious signal received from child.
```

30

tcxtest 31

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 SPARCstation5 and checks the FSV (fast SBus video) ASIC on the SPARCstation4 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.

`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*

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), etc.). 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 again.

This test does not perform any data verification, it just checks to see if any of these random accesses caused a time out.

constant

The `constant` 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

The `address` 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.

random

The `random` 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

The `blit` 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. It then `blits` the image to the screen. The destination images are read back and compared with the original image to verify the raw `blit` operation executed 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

The stip 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 (FSV), 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 SPARCstation4)

The cursor 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*

FB Locking

See the section about testing multiple frame buffers in *SunVTS 2.0 User's Guide* for details.

♦ **Click to enable or disable Frame Buffer locking.**

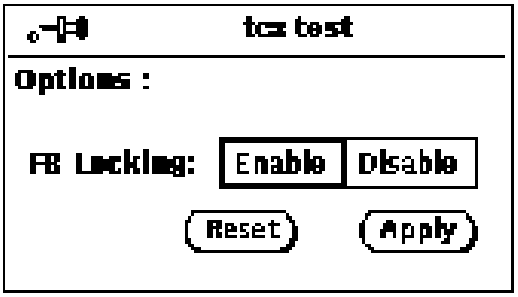


Figure 31-1 `tcxtest` Option Menu

`tcxtest` *Test Modes*

Connectivity Mode

Connectivity Mode is not applicable to frame buffer tests such as tcxtest.

Online Mode

Online Mode is not applicable to frame buffer tests such as tcxtest.

Offline Mode

Offline mode is the default for tcxtest. Tcxtest executes the appropriate subtests for the hardware platform being tested.

31

`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]*

Argument	Explanation
`dev=`*device_name*	Specifies the filename of the device to be tested, for example: **`dev=tcx0`**
`lock=`*E(nable)/D(isable)*	Enables / disables the window system locking option. See the section about Testing Frame Buffers in *SunVTS 2.0 User's Guide* for details. Do not use when device is the window system display.
`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 executed.
`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.

`tcxtest` *Error Messages*

The error messages below are returned by the various subtests.

```
6000 FBIOGATTR failed in is_24bit subroutine
```

Probable_Cause(s):

a. **Driver not installed**

b. **OS driver problem**

c. **afx bus error**

Recommended_Action(s):

a. **Examine system message files (/var/adm/messages) for other information**

```
6010 Colormap failure, Address: <address>, red, Expected:
<value> Observed: <value>
```

Probable_Cause(s):

a. **Bad RAMDAC**

b. **Bad a24 board**

c. **afx bus error**

Recommended_Action(s):

a. **Replace fram buffer card**

b. **Examine system message files (/var/adm/messages) for other information**

```
6011 Colormap failure, Address: <address>, green, Expected:
<value> Observed: <value>
```

Probable_Cause(s):

a. **Bad RAMDAC**

b. **Bad a24 board**

c. **afx bus error**

Recommended_Action(s):

a. **Replace frame buffer card**

b. **Examine system message files (/var/adm/messages) for other information**

```
6012 Colormap failure, Address: <address>, blue, Expected:
<value> Observed: <value>
```

Probable_Cause(s):

a. **Bad RAMDAC**

b. **Bad a24 board**

c. **afx bus error**

Recommended_Action(s):

a. **Replace fram buffer card**

b. **Examine system message files (/var/adm/messages) for other information**

```
6013 THC Cursor Regression failed Address: <address>,
Expected: <value>, Observed: <value>
```

Probable_Cause(s):

a. **Bad a24 board**

b. **afx bus error**

Recommended_Action(s):

a. **Replace frame buffer card**

b. **Examine system message files (/var/adm/messages) for other information**

```
6014 Can't clear frame buffer, dst_start: <address>,
src_start: <address>, Address: <value>, Observed: <value>
Expected: <value>
```

Probable_Cause(s):

a. **Bad a24 board**

b. **afx bus error**

Recommended_Action(s):

a. **Replace frame buffer card**

b. **Examine system message files (/var/adm/messages) for other information**

```
6015 Frame buffer pattern test failed, dst_space:
<address>, src_space: <address>, check_x = <value>, check_y
= <value>, Address: <value>, Observed: <value> Expected:
<value>
```

Probable_Cause(s):

a. **Bad a24 board**

b. **afx bus error**

Recommended_Action(s):

a. **Replace frame buffer card**

b. **Examine system message files (/var/adm/messages) for other information**

```
6016 Can't clear frame buffer pattern,  dst_space:
<address>, src_space: <address>, check_x: <value>, check_y:
<value>, Address: <value> Observed: <value> Expected:
<value>
```

Probable_Cause(s):

a. **Bad a24 board**

b. **afx bus error**

Recommended_Action(s):

a. **Replace frame buffer card**

b. **Examine system message files (/var/adm/messages) for other information**

```
6017 Can't clear frame buffer after write, Address:
<value>, Expected: <value> Observed: <value>
```

Probable_Cause(s):

a. **Bad a24 board**

b. **afx bus error**

Recommended_Action(s):

a. **Replace frame buffer card**

b. **Examine system message files (/var/adm/messages) for other information**

```
6018 Raw Blit test failed, Offset: <address>, Expected:
<value> Observed: <value>
```

Probable_Cause(s):

a. **Bad a24 board**

b. **afx bus error**

Recommended_Action(s):

a. **Replace frame buffer card**

b. **Examine system message files (/var/adm/messages) for other information**

```
6019 Blit test failed CHECKING, Offset: <address>,
Expected: <value> Observed: <value>
```

Probable_Cause(s):

a. **Bad a24 board**

b. **afx bus error**

Recommended_Action(s):

a. **Replace frame buffer card**

b. **Examine system message files (/var/adm/messages) for other information**

```
6020 Blit test failed, Offset: <value>, Expected: <value>
Observed: <value>
```

Probable_Cause(s):

a. **Bad a24 board**

b. **afx bus error**

Recommended_Action(s):

a. **Replace frame buffer card**

b. **Examine system message files (/var/adm/messages) for other information**

```
6021 Blit accel test failed. Offset: <value>, Expected
:<value> Observed: <value>
```

Probable_Cause(s):

a. **Bad a24 board**

b. **afx bus error**

Recommended_Action(s):

a. **Replace frame buffer card**

b. **Examine system message files (/var/adm/messages) for other information**

```
6022 Stipple test failed. Offset: <address>,
Expected:<value> Observed: <value>
```

Probable_Cause(s):

a. **Bad a24 board**

b. **afx bus error**

Recommended_Action(s):

a. **Replace frame buffer card**

b. **Examine system message files (/var/adm/messages) for other information**

```
6025 Raw Stipple test failed. Offset: <address>,
Expected:<value> Observed: <value>
```

Probable_Cause(s):

a. **Bad a24 board**

b. **afx bus error**

Recommended_Action(s):

a. **Replace frame buffer card**

b. **Examine system message files (/var/adm/messages) for other information**

```
6027 Frame buffer pattern test miscompare, y: <value> x:
<value> Observed: <value> Expected: <value> Address:
<value>
```

Probable_Cause(s):

a. **Bad RAMDAC**

b. **Bad a24 board**

c. **afx bus error**

Recommended_Action(s):

a. **Replace frame buffer card**

b. **Examine system message files (/var/adm/messages) for other information**

```
6029 Afx protocol test <message>  Expected: <value>
```

Probable_Cause(s):

a. **afx bus error**

b. **Bad a24 board**

c. **Bad cpu board**

Recommended_Action(s):

a. **Examine system message files (/var/adm/messages) for other information**

```
6030 Null allocation to membase
```

Probable_Cause(s):

a. **afx bus error**

b. **Bad a24 board**

c. **Bad cpu board**

Recommended_Action(s):

a. **Examine system message files (/var/adm/messages) for other information**

```
6031 Afx_mem test failure, Address: <address>, Expected:
<value>
```

Probable_Cause(s):

a. **afx bus error**

b. **Bad a24 board**

c. **Bad cpu board**

Recommended_Action(s):

a. **Examine system message files (/var/adm/messages) for other information**

```
6032 Afx_mem Mem test failure, Address+0x4000 Value:
<value>, Exp 0x5555aaaa
```

Probable_Cause(s):

a. **afx bus error**

b. **Bad a24 board**

c. **Bad cpu board**

Recommended_Action(s):

a. **Examine system message files (/var/adm/messages) for other information**

```
6033 Pattern test failed in Byte mode for DFB<value>.
Offset: <address>, Expected: <value> OBserved: <value>
xor(<value>)
```

Probable_Cause(s):

a. **afx bus error**

b. **Bad a24 board**

c. **Bad cpu board**

Recommended_Action(s):

a. **Examine system message files (/var/adm/messages) for other information**

```
6034 Pattern test failed in Short mode for DFB<value>.
Offset: <value>, Expected:  <value> OBserved: <value>
xor(<value>)
```

Probable_Cause(s):

a. **afx bus error**

b. **Bad a24 board**

c. **Bad cpu board**

Recommended_Action(s):

a. **Examine system message files (/var/adm/messages) for other information**

```
6035 Pattern test failed in Long mode for DFB<value>.
Offset: <value>, Expected: <value> OBserved:  <value>
xor(<value>)
```

Probable_Cause(s):

a. afx bus error

b. Bad a24 board

c. Bad cpu board

Recommended_Action(s):

a. Examine system message files (/var/adm/messages) for other information

```
6036 Pattern test failed in Long mode for DFB<value>
Offset: <value>, Expected: <value> OBserved:  <value>
```

Probable_Cause(s):

a. afx bus error

b. Bad a24 board

c. Bad cpu board

Recommended_Action(s):

a. Examine system message files (/var/adm/messages) for other information

```
6042 Pattern test failed in Dblword mode for DFB<value>,
Offset: <value>, Expected: <value> Observed:<value>,
xor:<value>
```

Probable_Cause(s):

a. afx bus error

b. Bad a24 board

c. Bad cpu board

Recommended_Action(s):

a. Examine system message files (/var/adm/messages) for other information

```
6039 Random test failed in Byte mode for DF<value>, Offset:
<value>, Expected: <value> Observed: <value> xor: <value>
```

Probable_Cause(s):

a. **afx bus error**

b. **Bad a24 board**

c. **Bad cpu board**

Recommended_Action(s):

a. **Examine system message files (/var/adm/messages) for other information**

```
6040 Random test failed in Short mode for DFB<value>,
Offset: <value>, Expected: <value> Observed: <value> xor:
<value>
```

Probable_Cause(s):

a. **afx bus error**

b. **Bad a24 board**

c. **Bad cpu board**

Recommended_Action(s):

a. **Examine system message files (/var/adm/messages) for other information**

```
6041 Random test failed in Long mode for DFB<value>,
Offset: <value>, Expected: <value> Observed: <value> xor:
<value>
```

Probable_Cause(s):

a. **afx bus error**

b. **Bad a24 board**

c. **Bad cpu board**

Recommended_Action(s):

a. Examine system message files (/var/adm/messages) for other information

```
6042 Pattern test failed in Dblword mode for DFB<value>,
Offset: <value>, Expected: <value> Observed: <value> xor:
<value>
```

Probable_Cause(s):

a. afx bus error

b. Bad a24 board

c. Bad cpu board

Recommended_Action(s):

a. Examine system message files (/var/adm/messages) for other information

```
6043 Address test failed in Char mode for DFB<value>.
Offset: <value>, Expected: <value> Observed: <value> xor:
<value>
```

Probable_Cause(s):

a. afx bus error

b. Bad a24 board

c. Bad cpu board

Recommended_Action(s):

a. Examine system message files (/var/adm/messages) for other information

```
6044 Address test failed in Short mode for DFB<value>,
Offset: <value>, Expected: <value> Observed: <value> xor:
<value>
```

Probable_Cause(s):

a. afx bus error

b. **Bad a24 board**

c. **Bad cpu board**

Recommended_Action(s):

a. **Examine system message files (/var/adm/messages) for other information**

```
6045 Address test failed in Long mode for DFB<value>,
Offset: <value>, Expected: <value> Observed: <value> xor:
<value>
```

Probable_Cause(s):

a. **afx bus error**

b. **Bad a24 board**

c. **Bad cpu board**

Recommended_Action(s):

a. **Examine system message files (/var/adm/messages) for other information**

```
6046 Address test failed in Dblword mode for DFB<value>,
Offset: <value>, Expected: <value> Observed: <value> xor:
value>
```

Probable_Cause(s):

a. **afx bus error**

b. **Bad a24 board**

c. **Bad cpu board**

Recommended_Action(s):

a. **Examine system message files (/var/adm/messages) for other information**

```
6047 Verifying Double:word writes,
```

Probable_Cause(s):

a. afx bus error

b. Bad a24 board

c. Bad cpu board

Recommended_Action(s):

a. Examine system message files (/var/adm/messages) for other information

`8000 Open failure for device <device name>`

Probable_Cause(s):

a. Driver not installed

b. OS driver prblm

c. afx bus err

Recommended_Action(s):

a. Examine system message files (/var/adm/messages) for other information

`8001 Pattern test failed in Long mode, Offset: <value>, Expected: <value> Observed: <value>`

Probable_Cause(s):

a. Driver not installed

b. afx bus error

c. Bad cpu board

Recommended_Action(s):

a. Examine system message files (/var/adm/messages) for other information

8002 can’t acquire console semaphor

Probable_Cause(s):

a. Not enough mem

b. Not enough swap space

Recommended_Action(s):

a. Examine system message files (/var/adm/messages) for other information

```
8003 XCreateSimpleWindow failed
```

Probable_Cause(s):

a. Not enough mem

b. Not enough swap space

Recommended_Action(s):

a. Examine system message files (/var/adm/messages) for other information

```
8004 Multibuffering extension does not exists.
```

Probable_Cause(s):

a. Not enough mem

b. Not enough swap space

Recommended_Action(s):

a. Examine system message files (/var/adm/messages) for other information

```
8005 Couldn’t create enough buffers.
```

Probable_Cause(s):

a. Not enough mem

b. Not enough swap space

Recommended_Action(s):

a. Examine system message files (/var/adm/messages) for other information

31

vmem 32

*Virtual Memory Test (*vmem*)*

This test 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 simulates a stressful environment for the operating system. This test also detects ECC parity errors, memory read errors, addressing problems, and displays the corresponding virtual memory addresses on failure.

Swap Space Consideration

Running this test puts a significant burden on the operating system, since it uses the majority of swap space available for testing. You should use the swap space reserve option to `vmem` when non-SunVTS test processes are started after SunVTS testing has started. See Swap Space Requirements in the *SunVTS 2.0 User's Guide* for a complete discussion of swap space requirements.

32

vmem *Options*

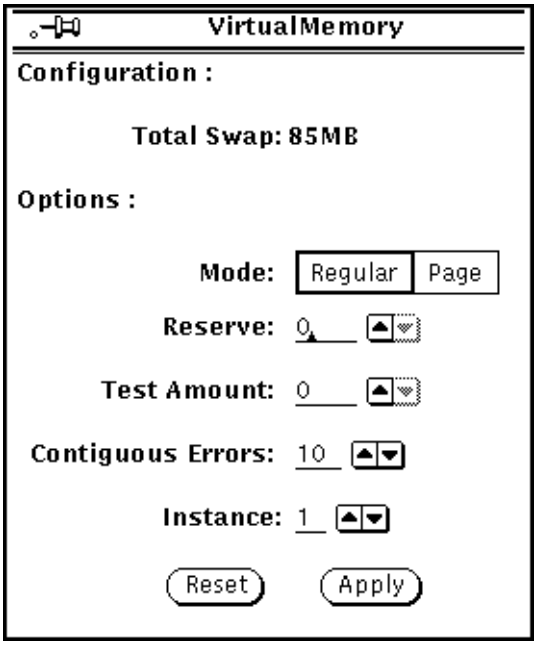


Figure 32-1 vmem Option Menu

vmem *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.

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 `vmem` or other running tests to fail.

Mode

There are two modes available:

Page mode tests assigned virtual memory one page at a time. Each page is mapped to the temporary file `/tmp/vmem.page` and then it is paged out to storage after the data is written. Next, the page is paged in before being read and compared.

Stress mode allocates (`valloc`) the entire assigned memory from the system, writing from beginning to end. Then the memory is read back and compared with the original pattern, one long word at a time.

In regular mode, the amount to be tested is limited by the physical memory size.

Reserve

The reserve option specifies the amount of memory to reserve from being tested by `vmem`. 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 was assigned to be tested by this instance, will cause the test to fail.

`vmem` *Test Modes*

Connectivity Mode

Not applicable for `vmem`.

Online Mode

Not applicable for `vmem`.

Offdline Mode

In this mode, `vmem` test 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.

32

vmem *Command Line Syntax*

/opt/SUNWvts/bin/vmem *standard_arguments* **-o mode=***type*, **reserve=***number*, **amount=***number*, **bdinfo**=*number*, **cerr**=*number*

Argument	Explanation
mode=*type*	Specifies which mode of the vmem test you want to run; you can choose between page mode, which tells the write/read memory test to proceed one system memory page at a time, and regular, which valloc's the entire assigned memory which is read and compared, one long word at a time
reserve=*number*	Specifies the amount of Mbytes of virtual memory to reserve in addition to the default amount
amount=*number*	Specifies the number of Mbytes of memory to be tested instead of the default amount
bdinfo=*number*	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).
cerr=number	Specifies the maxiumu number of contiguous errors to be dumped when a memory error occurs.

vmem *Error Messages*

Note – 6001-6004 and 8001-8010 are fatal errors.

```
6001  setrlimit(RLIMIT_DATA) failed to set to
RLIM_INFINITY: <error_message>
```

Probable_Cause(s):

a. Failed to increase the size of test process' heap to 2GB.

```
6002  Test terminated after finding maximum number
(<number>) of noncontiguous errors.

6003  msync() failed: <error_message>
```

Probable_Cause(s):

a. Failed to synchronize memory with physical storage.

6004 Test terminated after finding miscompared data.

Probable_Cause(s):

a. Test failed.

8001 Test completed. Found %d noncontiguous miscompare errors.

Probable_Cause(s):

a. Suspect swap partition(s).

b. Suspect SCSI controller.

c. Faulty system software.

Recommended_Action(s):

a. Consult your authorized Sun service provider.

8002 Increase swap space to allocate atleast <*number*> MB from the process' heap.

Probable_Cause(s):

a. Failed to increase mem **process' heap storage.**

8003 Attempt to reserve more than what's been assigned.

Recommended_Action(s):

a. Reduce the "reserve" amount.

8004 valloc(<address>) failed: <error_message>

Recommended_Action(s):

a. Increase swap space or reduce the reserve amount.

8005 Open(<file_name>) failed: <error_message>

Probable_Cause(s):

a. Failed to open the file <file_name> used in the "page" mode.

```
8006  mmap() virtual address <address> to <file_name>
failed: <error_message>
```

Probable_Cause(s):

a. Failed to map a region at the virtual address of the file <file_name> used in the "page" mode.

```
8007  munmap(<address>) failed: <error_message>.
```

Probable_Cause(s):

a. Failed to unmap a region at the virtual address of the file <file_name> used in the "page" mode.

```
8008  close() failed on <file_name>: <error_message>

8009  Not enough space to store miscompare information.
```

Probable_Cause(s):

a. Failed to allocate space to store miscompare data.

Recommended_Action(s):

a. Increase swap space or reduce the load on the system.

```
8010  swapctl() failed: <error_message>
```

Probable_Cause(s):

a. Failed to obtain the amount of swap space configured in the system.

xbtest 33

*SBus Expansion Subsystem (*xbtest*)*

This test verifies the functionality of the Sun SBus Expansion Subsystem (XBox) hardware and its peripherals. The SBus Expansion Subsystem can be operated in two mutually exclusive modes: transparent and nontransparent. Transparent mode test checks SBus Expansion Subsystem peripherals, such as SBus cards and disk drives. Do not select transparent mode test if the SBus Expansion slots are empty. Nontransparent mode test checks the actual Expansion Subsystem itself.

Note – Do not select nontransparent mode test if SBus cards are installed in the SBus Expansion slots.

Transparent Mode

In transparent mode, `wait_for_error` test is used. When `xbtest` is invoked, a child process is created. The parent process waits for the child process to terminate and then it exits. The child process issues a `WAIT_FOR_ERROR_PAK ioctl` call and then enters sleep mode.

When the device driver receives an error, it wakes up and passes an error packet to the child process. The child process dumps the contents of the error packet, and also exits.

If the child process does not receive an error message before the end of the test period spevified by the **time** option, it exits. The parent process then exits. The default value for the **time** option is one minute. If you do not specify another value for the **time** option, the default value is used in both regular and stress test modes.

Nontransparent Mode

In nontransparent mode, `self_diagnostic` test is used. If you do not have an XBox SBus card in any slot of your system, you must add the following line to the `/etc/system` file:

```
set xbox:xbox_no_cards_in_slot0=1
```

After adding this line to `/etc/system`, reboot the machine using the **-r** option. You can now run the nontransparent mode test of `xbtest`.

This test checks the Expansion Subsystem hardware; the subtests are described below. The subtests, part of the diagnostic test, are repeated three times in both regular and stress test modes. Time-out checks are included to avoid indefinite hangs. Failure of any test should not result in a system panic, although full recovery is not guaranteed.

Self Diagnostic Test

Follow these steps to do a self-diagnostic test:

1. Do a hard reset.
2. Check for the expected value from XAC register.
3. Do a DVMA XAC Interrupt Test:
   a. Cause a DVMA transfer by asserting DVTE + INTT in control register 1 of XAC.
   b. Wait for the interrupt.
   c. Compare the error status packet with the expected values.
4. Do a DVMA XBC Interrupt Test:
   a. Cause a DVMA transfer by asserting DVTE + INTT in control register 1 of XBC.

b. Wait for the interrupt.

c. Compare the error status packet with the expected values.

`xbtest` *Options*

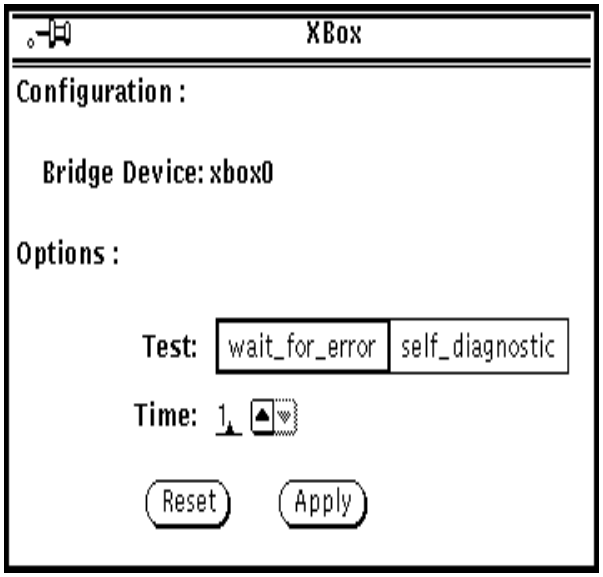


Figure 33-1 `xbtest` Option Menu

The top section of the `xbtest` Options menu displays the device name of the Sbus Expansion Subsystem card being tested.

- ***Test*** specifies either `self_diagnostic` or `wait_for_error` test. See “Nontransparent Mode” on page 398 before attempting to run the `self_diagnostic` test.
- ***Time*** specifies how long this xbtest waits for an error to be returned. You can change the `xbtest` time setting by selecting the text field and typing the number of minutes you want the test to run (mininum 1 minute; maximum 20 minutes).

33

xbtest *Test Modes*

Connectivity Mode

Not available for xbtest.

Online Mode

Not available for xbtest.

Offline Mode

A wait time of 10 seconds is used with the wait_for_error test if the XBox is in transparent mode. If the XBox is in nontransparent mode, the subtests which are part of the self-diagnostic test are repeated three times.

xbtest *Command Line Syntax*

/opt/SUNWvts/bin/xbtest -o dev=*xbox#*,**test=***we | sd*,**time=***test_time*

Argument		Explanation
dev=*xbox#*		Specifies the physical path name of the subsystem unit to be tested (*.xbox#*), where # is the instance number defined in path_to_inst(4). You need to search the /devices tree to find the physical pathname of the subsystem. **Note:** You must include the *device_name* path when running xbtest from the command line.
time=*test_time*		Specifies how long this test waits for an error to be returned. Used only with the *we* option, Substitute *test_time* with the number of minutes you want xbtest to wait. The default is one minute.
test=*we	sd*	*we*	Specifies wait_for_error test. This option directs xbtest to run in transparent mode. This option runs xbtest continually until an error is returned, or until the time interval specified with the T=*test_time* option has ended.
	sd	Specifies self_diagnostic test. This option is used in nontransparent mode. See "Nontransparent Mode" on page 398, before attempting to run nontransparent mode.

For example, to run xbtest, type:

```
# /opt/SUNWvts/bin/xbtest -o dev=xbox0,test=we,time=2
```

xbtest *Error Messages*

6002 Incorrect options

6003 XAC_WAIT_FOR_ERROR ioctl command fail (<error_message>).

6004 XAC_CLEAR_WAIT_FOR_ERROR ioctl command fail (<error_message>).

6005 XAC_WRITE0 ioctl command fail (<error_message>).

6006 Failed to open /etc/path_to_inst file (<error_message>).

6007 <device_name> invalid device name.

6008 <device_name>’s instance number(%d) not in /etc/path_to_inst file.

6009 error status dirty bit <number>.

Probable_Cause(s):

a. **This bit is set to 0 when the XBox detects an error.**

6010 error status indicator <number>.

Probable_Cause(s):

a. **This bit is set to 0 when the XBox detects an error.**

6011 expansion sbus slave selects 0x%x.

6012 parent master request %d.

```
6013  packet type %d.

6014  physical address info %d.

6015  sbus size %d.

6016  cable resend limit timeout error (dpr1).
```

Probable_Cause(s):

a. This error is detected by the XBox controller.

```
6017  cable parity error (dpr1).
```

Probable_Cause(s):

a. This error is detected by the XBox controller.

```
6018  expansion sbus read error (err ack).
```

Probable_Cause(s):

a. The signal err ack is asserted on the expansion SBus by XBox slave while being read by an XBox master.

```
6019  expansion sbus read error (rsvd ack).
```

Probable_Cause(s):

a. The signal rsvd ack is asserted on the expansion SBus by XBox slave while being read by an XBox master.

```
6020  expansion sbus read error (late error).
```

Probable_Cause(s):

a. The signal rsvd ack is asserted on the expansion SBus by XBox slave while being read by an XBox master.

```
6021  expansion sbus timeout error.
```

Probable_Cause(s):

a. XBox controller detects a timeout on the expansion SBus during a read or write to an XBox slave..

```
6022  write 0 error.
```

```
6023  buffer write error (err ack).
```

Probable_Cause(s):

a. The signal err ack is asserted on the expansion or host SBus while the XBox controller / XAdapter was performing a DVMA write.

```
6024  buffer write error (rsvd ack).
```

Probable_Cause(s):

a. The signal rsvd ack is asserted on the expansion or host SBus while the XBox controller / XAdapter was performing a DVMA write.

```
6025  buffer write error (late error).
```

Probable_Cause(s):

a. The signal late error is asserted on the expansion or host SBus while the XBox controller / XAdapter was performing a DVMA write.

```
6026  cable resend timeout error (dpr0).
```

Probable_Cause(s):

a. This error is detected by the XAdapter controller.

```
6027  cable ack timeout error.
```

Probable_Cause(s):

a. This error occurs when XAdapter does not detect cable acknowledgement.

```
6028  cable parity error (dpr0).
```

Probable_Cause(s):

a. This error is detected by the XAdapter controller.

```
6029  cable serial interrupt parity error.
```

Probable_Cause(s):

a. This error is detected by the XAdapter controller.

```
6030  child not ready error.
```

Probable_Cause(s):

a. This error indicates that the XBox controller no longer can communicate with the XAdapter.

```
6031  XAC_GET_REG_VALUES ioctl command fail
(<error_message>).

6032  XAC_GET_ERROR_PKT ioctl command fail
(<error_message>).

6033  XAC_TRANSPARANT ioctl command fail (<error_message>).

6034  XAC_NON_TRANSPARANT ioctl command fail
(<error_message>).

8003  XAC_RESET ioctl command fail (<error_message>).

8004  XAC_REG_CHECK ioctl command fail (<error_message>).
```

Loopback Connectors A

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 Pin Connections for Loopback Plugs

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

Table A-1 Pin Connections for Loopback Plugs *(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
Clear To Send (CTSa)	CB	106	9	27	2	8	5	input	CB
Data Set Ready (DSRa)	CC	107	11	29	NC	6	6	input/ output	CC
Signal Ground (SG)	AB	102	19	NC	4	5	7	none	AB
Data Carrier Detect (DCDa)	CF	109	13	31	7	1	8	input	CF
Transmit Clock In (TRxCa)	DB	114	5	23	NC	NC	15	input	DB
Receive Clock in (RTxCa)	DD	115	8	26	8	NC	17	input	DD
Data Terminal Ready (DTRa)	CD	108	12	30	1	4	20	output	CD
External Clock Out (TRxCa)	DA	113	17	35	NC	NC	24	output	DA
Secondary Data Carrier Detect (DCDb)	SCF	122	NC	NC	NC	NC	12	input	SCF
Secondary Clear to Send (CTSb)	SCB	121	NC	NC	NC	NC	13	input	SCB
Secondary Transmit Data (TxDb)	SBA	118	NC	NC	NC	NC	14	output	SBA
Secondary Receive Data (RxDb)	SBB	119	NC	NC	NC	NC	16	input	SBB
Secondary Request to Send (RTSb)	SCA	120	NC	NC	NC	NC	19	output	SCA

Notes: 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. The wiring is shown in Figure A-1.

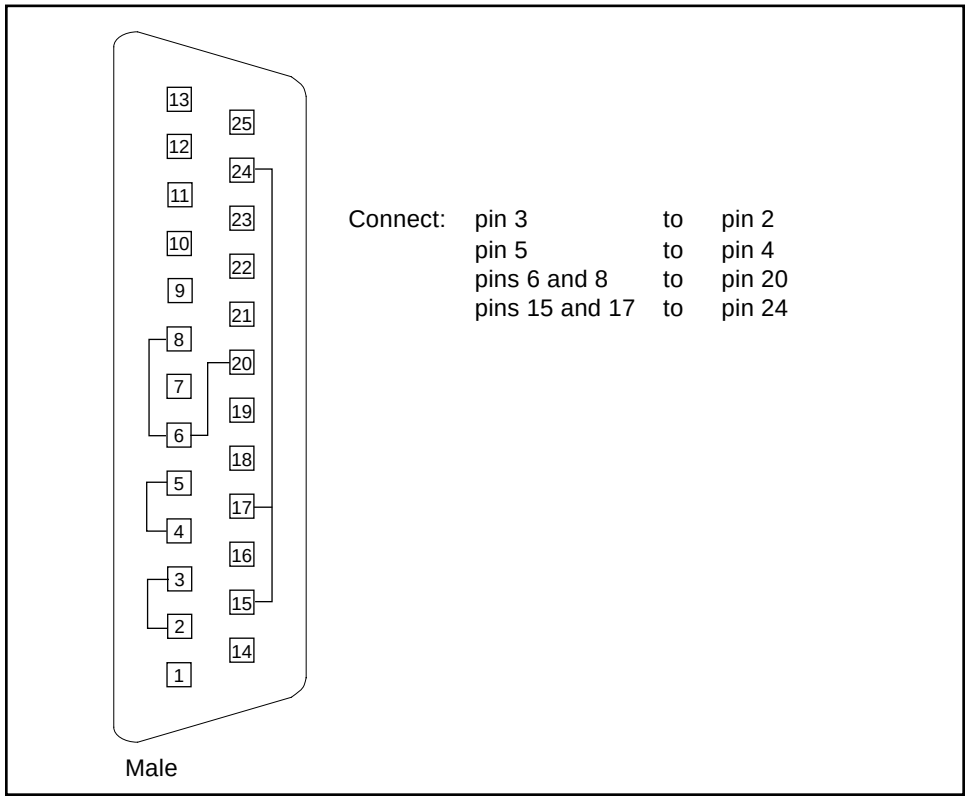


Figure A-1 25-pin RS-232 Loopback Plug

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

Use these wiring directions 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. The wiring is shown in Figure A-2.

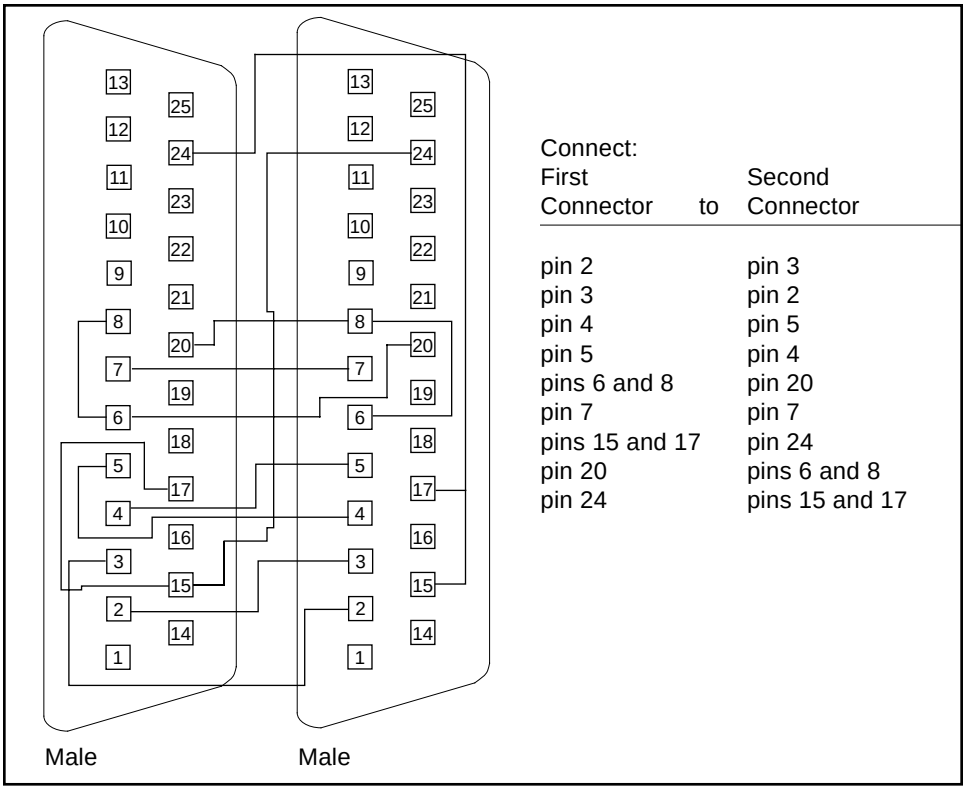


Figure A-2 25-pin RS-232 Port-to-Port Loopback Cable

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 (see Figure A-3). Both connectors are male.

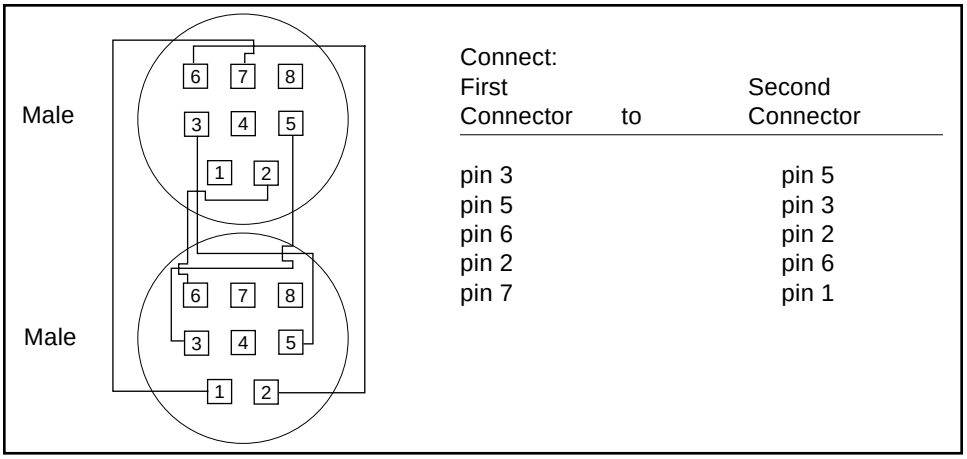


Figure A-3 8-Pin to 8-Pin Loopback Cable

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 (see Figure A-4).

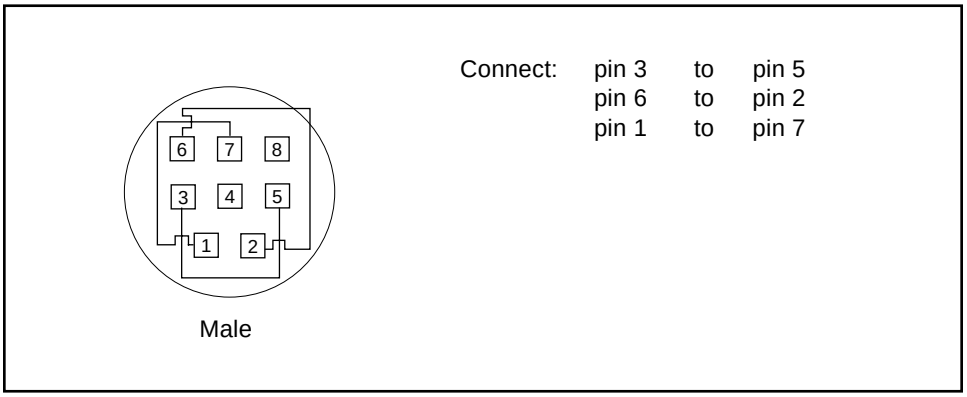


Figure A-4 8-Pin Loopback Plug

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

A

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 (see Figure A-5).

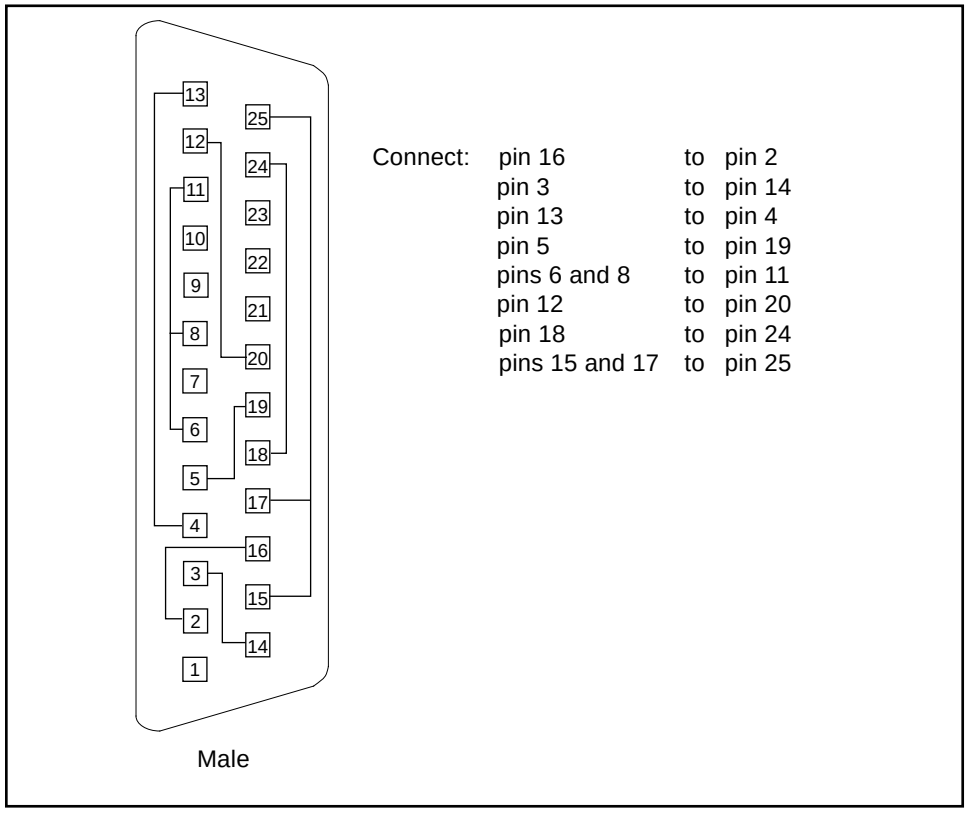


Figure A-5 Port A-to-Port B Loopback Plug

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 (see Figure A-6):

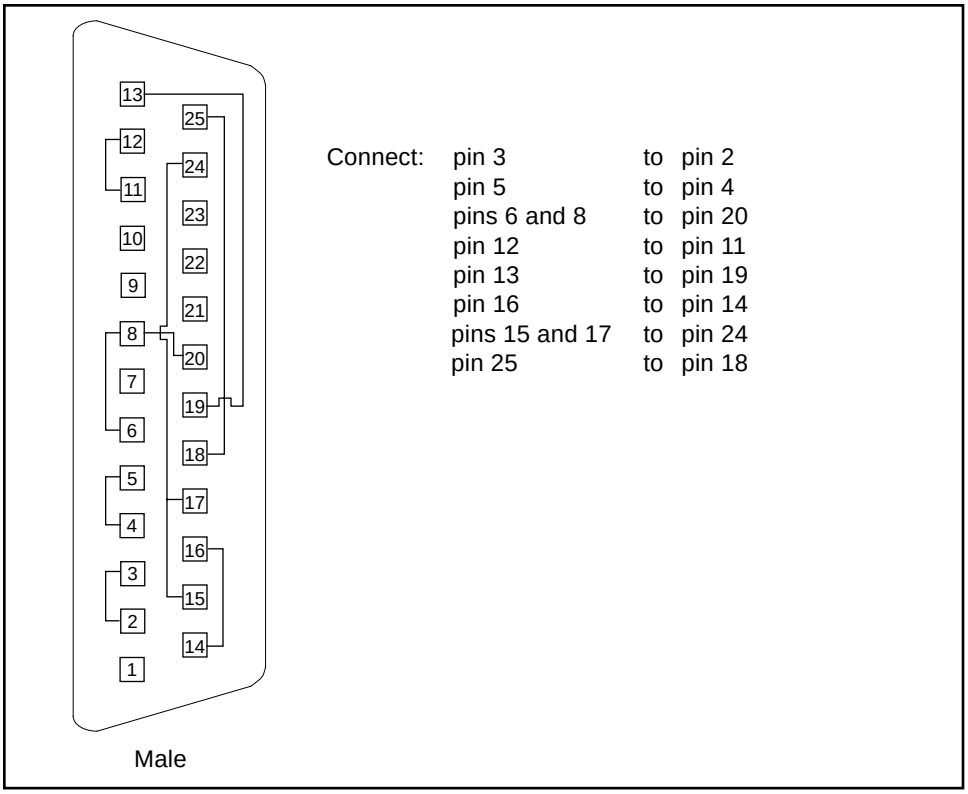
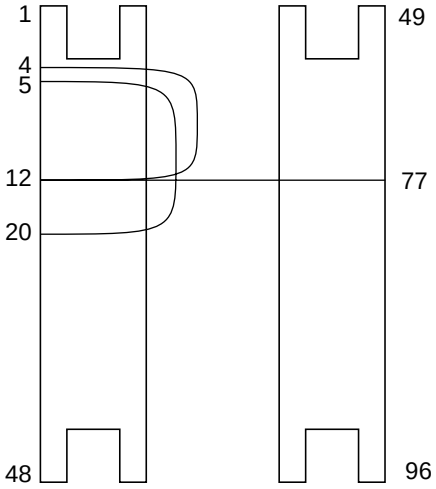


Figure A-6 Port A-to-A, Port B-to-B Loopback Plug

A.1 96-Pin Female Loopback Connector

This 96-pin connector (see Figure A-7) can be ordered from Sun Microsystems (Part Number 370-1366).



Materials:

PCR-E96FA	(1)
PCS-E96LKPA	(1)
3751 Metal Plug (9563K42)	(1)
AWG28 Madison Cable UL/CSA Approved	(8” long)

Connect:

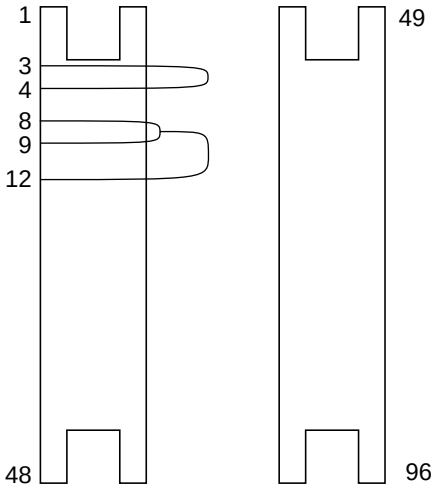
pins 4 and 12	to	pin 77
pin 5	to	pin 20
pin 6	to	pin 36
pin 7	to	pin 44
pin 10	to	pin 11
pin 13	to	pin 16
pin 14	to	pin 15
pin 18	to	pin 19
pin 21	to	pin 24
pin 28	to	pin 60
pin 29	to	pin 68
pin 30	to	pin 34
pin 37	to	pin 40
pin 38	to	pin 39
pin 42	to	pin 43
pin 45	to	pin 48
pin 46	to	pin 47
pin 52	to	pin 78
pin 53	to	pin 55
pin 54	to	pin 75
pin 58	to	pin 59
pin 61	to	pin 64
pin 62	to	pin 63
pin 66	to	pin 67
pin 69	to	pin 72
pin 76	to	pin 79
pin 82	to	pin 83
pin 85	to	pin 88
pin 86	to	pin 87
pin 90	to	pin 91
pin 93	to	pin 96
pin 94	to	pin 95

Figure A-7 96-Pin Female Loopback Connector

A.2 96-Pin Female Special Loopback Connector

This 96-pin connector (see Figure A-8) can be ordered from Sun Microsystems (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)	

Connect:

pin 3	to	pin 4
pin 5	to	pin 7
pins 8 and 9	to	pin 12
pin 10	to	pin 11
pin 13	to	pin 14
pin 15	to	pin 17
pins 18 and 19	to	pin 22
pin 20	to	pin 21
pin 27	to	pin 28
pin 29	to	pin 31
pins 32 and 33	to	pin 36
pin 34	to	pin 35
pin 37	to	pin 38
pin 39	to	pin 41
pins 42 and 43	to	pin 46
pin 44	to	pin 45
pin 51	to	pin 52
pin 53	to	pin 55
pins 56 and 57	to	pin 60
pin 58	to	pin 59
pin 61	to	pin 62
pin 63	to	pin 65
pins 66 and 67	to	pin 70
pin 68	to	pin 69
pin 75	to	pin 76
pin 77	to	pin 79
pins 80 and 81	to	pin 84
pin 82	to	pin 83
pin 85	to	pin 86
pin 87	to	pin 89
pins 90 and 91	to	pin 94
pin 92	to	pin 93

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

Figure A-8 96-Pin Female Special Loopback Connector

A.3 37-Pin RS-449 Loopback Cable

Use these wiring instructions for a loopback cable for two 37-pin RS-449 synchronous ports (see Figure A-9).

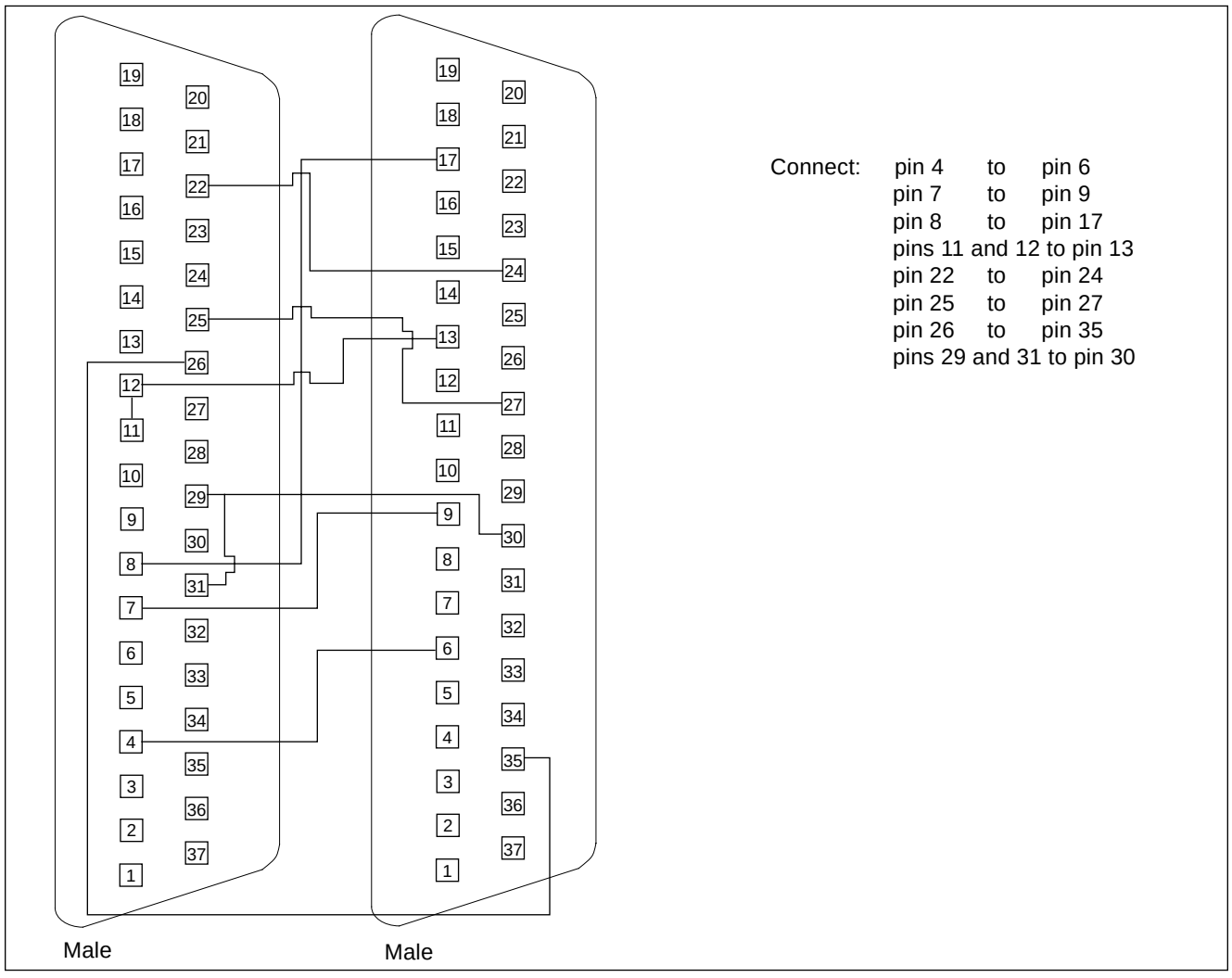


Figure A-9 37-Pin RS-449 Loopback Cable

A.4 37-Pin RS-449 Loopback Plug

Use these wiring instructions for making a male 37-pin RS-449 loopback plug (see Figure A-10). This connector is also available from Sun (Part Number 530-1430).

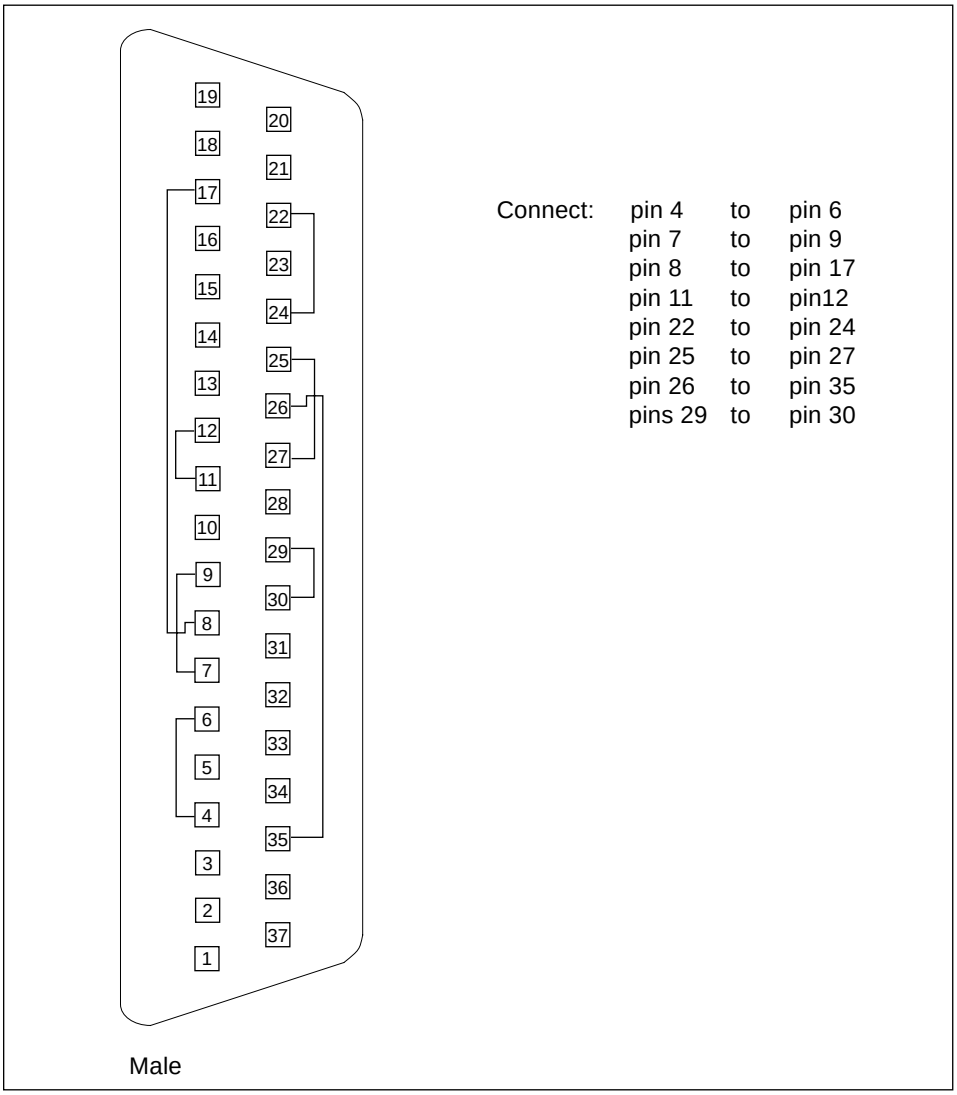


Figure A-10 37-Pin RS-449 Loopback Plug

A.5 9-pin Male Single-port Loopback Plug

Use these wiring directions for male 9-pin RS-232 and RS-423 single-port loopback plugs (see Figure A-11):

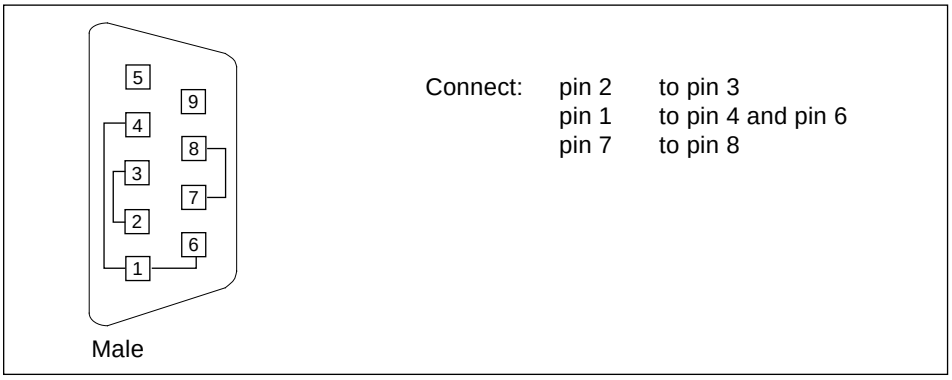


Figure A-11 9-Pin Male Single-port Loopback Plug

A.6 9-pin Female Single-port Loopback Plug

Use these wiring directions for female 9-pin RS-232 and RS-423 single-port loopback plugs (see Figure A-12):

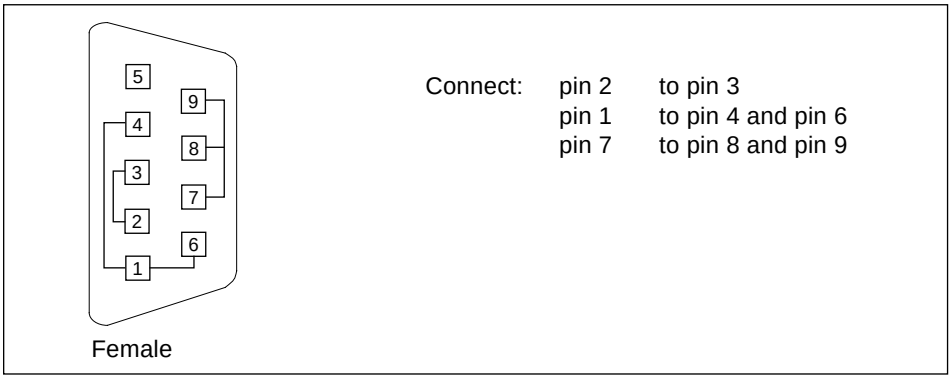


Figure A-12 9-Pin Female Single-port Loopback Plug

Note – Use this loopback plug with the `pcmciatest`.

A.7 9-Pin to 25-Pin Port-to-Port Loopback Cable

Use these wiring directions for a 9-pin RS-232 and RS-423 port to 25-pin RS-232 and RS 423 port loopback cables (see Figure A-13). Both connectors are male.

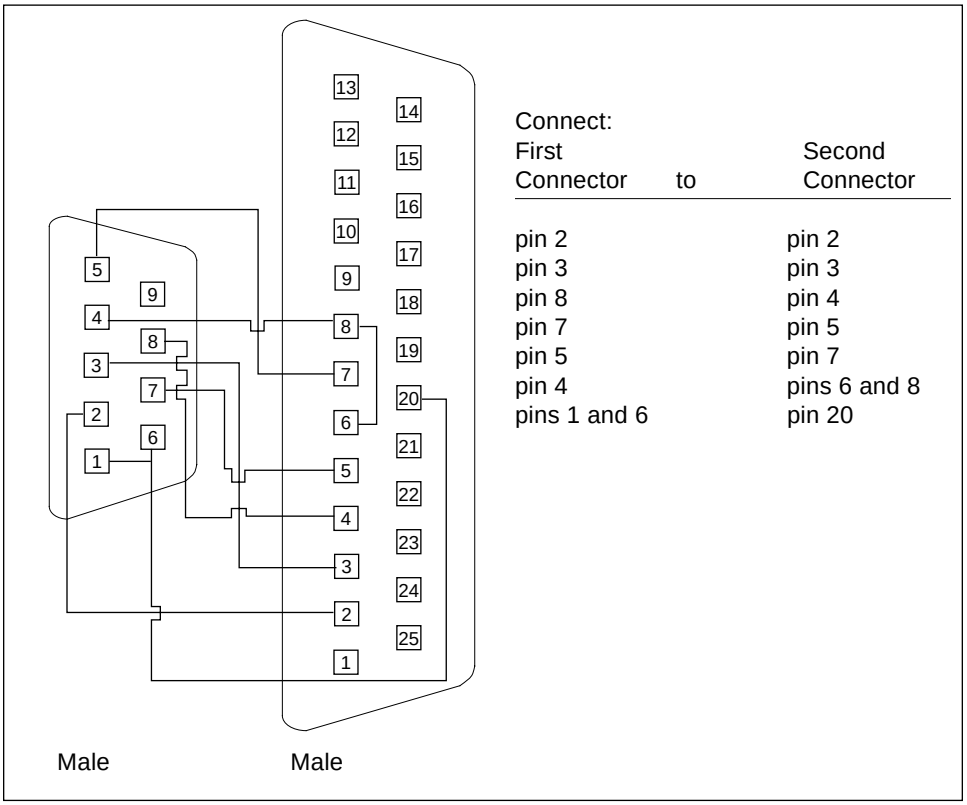
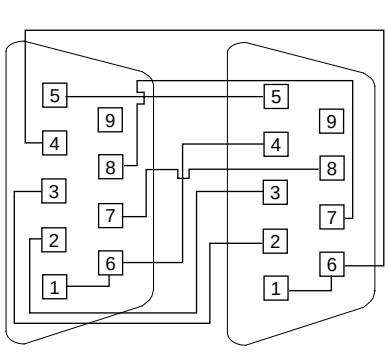


Figure A-13 9-Pin to 25-Pin Port-to-Port Loopback Cable

A.8 9-Pin to 9-Pin Port-to-Port Loopback Cable

Use these wiring directions for 9-pin RS-232 and RS 423 port to 9-pin RS-232 and RS-423 port loopback cables (see Figure A-14). 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

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 pin1 to pin1, pin 2 to pin 2, and so on, for all pins. This loopback is a “straight-through” connection.

Index

Numerics

25-pin loopback plug, 303
25-pin port A-to-A port B-to-B loopback plug, 410
25-pin port A-to-port B loopback plug, 410
25-pin RS-232 loopback plug, 407
25-pin RS-232 port-to-port loopback cable, 407
8-pin loopback plug, 409
8-pin to 8-pin loopback cable, 408
96-pin loopback plug, 303

A

acceleration tests, 369
accelerator, 275
access
  read / write, 275
access check
  file I / O, 275
accessing SunVTS, 1
address space
  mapped, 275
AFX protocol tests, 369
AnswerBook
  on-line documentation, xxvi
application-specific integrated circuit, (ASIC), 282
arguments
  command line, 2
  generic, 2
  standard, 2
  test-specific, 2
ASCII character set, 304
ASIC, (application-specific integrated circuit
  integrated circuit, 282
audio, 20
  distorted, 10
  error
    messages, 20
  jacks, unused, 16
  messages, error, 20
  not audible, 10
  option menu, 10
  options, 10
  output, 17
  ports, 10
  subtests
    controls, 9
    crystal, 9
    loopback, 9
    record / play, 9
  test
    music play, 16

test modes, 17
volume
music play test, 17
audio test, 9
audioamd(7) man page, 10
audiocs(7) man page, 10
available
test modes, 4

B

banks of memory
memory
banks, 270
battery power, 275
baud rate, 304, 306
bi-directional parallel port printer
test, 231
bidirectional printer test, 25
blits
screen test, using, 56
block of data, 275
board
battery check, 275
board memory, 275
board performance check, 275
bpptest, 25
arguments, 28
bidirectional parallel port printer
test, 25
command line syntax, 28
error messages, 28
option
access, 26
mode, 26
options, 26
printer test hardware and software
requirements, 26
test modes, 27
buffer
memory test, 56
buffer test mode, 55

C

cache
memory, 275
card
SunVideo, 284
CD
cdtest, 31
scratch, 31
test, 31
type, 31
CDE (Common Desktop Environment), 1
CDE, (common desktop
environment), xxv
cdtest
arguments, 34
command line syntax, 33, 34
compact disc test, 31
error messages, 35
not scalable, 31
options, 32
audio test, 32, 33
CD type, 32
percent data/track, 32
read mode, 32, 33
volume, 32, 33
test modes, 33
cg14test, 42
arguments, 48
color graphics frame buffer test, 39
command line syntax, 48
data bus test, 42
error messages, 49
groups
CG14 display, 40
DAC chip registers, 40
data propagation, 40
driver, 41
driver IOCTLs, 40
hardware, 40
MDI and VBC chip control
registers, 40
MDI chip CLUT registers, 40
MDI chip cursor registers, 40
MDI chip testmode, 40

MDI chip XLU registers, 40
memory chips, 40
visual pattern, 40
options, 47
FB locking, 47
long test, 47
P-affinity, 47
registers
control, 45
MDI and VBC chip control, 41
MDI chip cursor, 44
MDI chip XLUT, 45
RAMDAC, 45
restrictions, 39
test modes, 49
tests VSIMM devices, 39
VRAM data retention, 43
write recovery test for EMC mapping, 43
`cg14test` cg14 framebuffer test, 39
cg6
framebuffer test, 55
options, 57
cg6test
command line syntax, 58
error messages, 58
frame buffer, GX, GX+, and TGX options test, 55
mode, offline, 57
options
blit test, 56
colormap test, 56
cursor test, 55
fast copy in double buffer test mode, 55
FB locking, 57
FBC test, 56
frame buffer test, 56
line test, 56
polygon test, 56
screen test using blits, 56
TEC test, 56
character length, 304
circuitry
SBus DMA, 25
CL4000
compression engine, 282
CL4000, rtvctest, 281
color graphics frame buffer test (cg14test), 39
colormap test
cg6, 56
colormap tests, 370
command line
syntax, 2
command line arguments
-e, 3
-i number, 3
-r, 2
-s, 2
standard, 2
-u, 2
-w number, 3
command line syntax
bpptest, 28
cdtest, 34
cg14test, 48
ecpptest, 96
fbtest, 104
isdntest, 210
mptest, 244
command-line arguments, 2
-c, 2
-d, 3
-t, 3
commands
software, xxvi
UNIX, xxvi
comment
card, xxviii
email or FAX, xxviii
reader, xxviii
common desktop environment (CDE), xxv
Common Desktop Environment, (CDE), 1
compact disc test
cdtest, 31
compress
video data, 282

compression engine
CL4000, 282
concurrent
tests, 276
configuration
hardware, 301
connectivity mode, 4
control
logic, 56
controller
board, 263
networking, 251
SSA card, 266
core dump, enabled, 2
CPU tests
`cg14test`, 39
`cg6` cgsix framebeuffer test, 55
multiprocessing test `mptest`, 241
network test `nettest`, 251
`tcxtest` S24 framebuffer test, 369
cursor tests, 370
customtest, 281

D

data
blocks, 275
buffered, 263
data pattern, 27
data type pattern, selectable, 304
dbri(7) man page, 10
debug mode, 3
default
print interval, 27
default location, for OPEN LOOK files, 6
delimiters, 266
device
driver, SBus, 25
device driver
printer test, 94
devices
networking, 251
digitize
video data, 282
disc, 31
disk drives
logical, 264
disktest, 69
command line syntax, 75
error messages, 76
test modes, 74
test options, 71
display usage statement, 2
distorted
audio, 10
DMA circuitry, 25
documentation
Solaris 2.x, xxvi
documents
related to SunVTS, xxvii
dots per inch (dpi), 292
double buffer test mode, 55
double buffering capacity, 55
dual basic rate ISDN (DBRI) chip (isdntest), 205

E

ECC (error correction code), 269
echo-TTY, 304
ECP 1284
parallel port printer test, 93
ecpp
device, 93
driver, 93
hardware requirements, 93
software requirements, 93
ecpptest
command line syntax, 96
error messages, 96
hardware and software requirements, 93
options, 95
parallel port printer test, 93
subtests
external passive loopback, 94
internal test fifo loopback, 94

printer test, 94
test modes, 95
email
comments, xxviii
EMC mapping access modes, 43
environment
OPEN LOOK, 1
error
correction code (ECC), 269
error messages
bpptest, 28
cdtest, 35
cg14test, 49
ecpptest, 96
fatal, range of, 394
fbtest, 105
isdntest, 211
mptest, 245
nettest, 255
plntest, 267
pmem, 271
pstest, 277
spdtest, 295
spif, 306
sptest, 315
sunbuttons, 321
sundials, 325
sxtest, 348
systest, 352
tapetest, 363
tcxtest, 375
vmem, 394
xbtest, 401
error mode, 2
errors
memory, 269
parity, 269
ethernet test, 251
external loopback test, 93
external passive loopback test, 94

F

fan module
plntest, 264
fast frame buffer test (ffbtest), 109
fatal error messages, range of, 394
FAX
comments, xxviii
FBC test
cg6 option, 56
fbtest
command line syntax, 104
error messages, 105
test modes, 104
`fbtest` framebuffer test, 103
fbtest, frame buffer test, 103
ffbtest, 109
3DRAM logic test description, 111
3DRAM test description, 111
command line syntax, 117
error messages, 118
options, 110
RAMDAC test description, 112
test modes, 115
fiber optic test, 251
file
I/O access check, 275
files, 25
raster, 25
floating point unit test (fputest), 125
flow control, selectable, 304
fputest, 125
command line syntax, 127
error messages, 127
options, 126
test modes, 126
frame buffer
card, 39
generic test, fbtest, 103
locking, 57
multiple, 7
test, 39
testing, 7
frame buffer test
tcxtest, 369
frame buffers

double buffering capacity, 55
FRUs
identified by spif, 306
function trace mode, 3

G

generic frame buffer test, fbtest, 103
getsubopt(3c) man page
gives test-specific argument format, 3
graphical user interface (GUI), xxv
graphical user interfaces (GUIs), 1
GUI (grapical user interface), xxv
GX (graphics) option test, 55

H

hardware
configuration, 301
CPU board, networking, 251
requirements, spif test, 301
setup, 6
verification, 6
host
system, 263

I

I/O
file access check, 275
ICMP (Internet control message protocol, 251
incremental test
nettest, 252
instance
assigned for scalable test, 3
instances
number of, 3
interface
command line, 2
TTY, 2
Internet control message protocol (ICMP), 251
intervals
printing, 27
intervals, of print images, 293
intervention mode
spif test, 301
isdntest, 205
command line syntax, 210
error messages, 211
options, 209
test modes, 210
`isdntest` ISDN test, 205

J

Jalapeno, rtvctest, 281

L

leotest, 213
command line examples, 222
command line syntax, 221
error messages, 222
options, 214
subtests, 216
test modes, 220
library
location, 6
line test
cg6, 56
logic
frame buffer controller (FBC), 56
transformation engine and cursor (TEC) control, 56
logical
disk drives, 264
loopback, 303
cable, 17
test, 17
test type, 17
loopback connectors
25-pin port A-to-A port B-to-B plug, 410
25-pin RS-232 plug, 407
25-pin RS-232 port-to-port cable, 407
8-pin plug, 409
8-pin to 8-pin cable, 408

pin connections, 405
port A-to-port B plug, 410
See Appendix A
see Appendix A, 303
loopback plug, 303
loopback settings, table of, 312
loopback test
external, 93
lpvitest, 231
command line syntax, 234
error messages, 234
hardware and software requirements, 231
options, 232
test mode, 233
verifies SPARCprinter functionality, 25

M

magnets
SPARCprinter paper tray, 26
man page
audioamd(7), 10
audiocs(7), 10
dbri(7), 10
mapped
address space, 275
memory
board, 275
cache, 275
frame buffer, 56
physical, 269
rtvctest, 281
virtual, 392
memory tests
physical memory `pmem`, 269
virtual memory `vmem`, 391
menu
bpptest option, 26
messages
bpptest error, 28
range of fatal errors, 394
mode
debug, 3
error, 2
stress, 3
trace, 3
verbose, 2
monitor
greyscale, 56
monochrome, 56
mouse
movement, can cause test failure, 7
mptest, 241
command line syntax, 244
error messages, 245
options, 242
test modes, 244
`mptest` multiprocessing test, 241
multiple packet phase, 310
multiprocessor test (mptest), 241
music
play test, 16

N

nettest
command line syntax, 255
error messages, 255
options
number of retries field, 254
print warning field, 254
subtest
incremental test, 252
pattern test
pattern test
nettest, 252
random test, 252
test modes, 254
`nettest` network hardware test, 251
nettest options
receive timeout field, 253
target host field, 253
networking
controller, 251
devices, 251
hardware on CPU board, 251

NeWSprinter test, 291
spdtest, 291
NFS accelerator, 275
nonvolatile RAM, 263
NTA patterns, 42
number of retries field
nettest, 254
NVRAM
battery module, 264

O

offline mode, 4
on-line
AnswerBook, xxvi
online mode, 4
OPEN LOOK, xxv
default file location, 6
environment, 1
user interface, 1
version, 6
option menu
sxtest, 334
options
audio, 10
bpptest, 26
cdtest, 32
cg14test, 47
cg6, 57
isdntest, 209
mptest, 242
pmem, 269
pstest, 276
rtvctest, 284
spdtest, 292
sptest, 311
systest, 351
tapetest, 355
tcxtest, 373
vmem, 392
xbtest, 399
ordering
Sun documents, xxviii
overlay registers
RAMDAC, 55

P

packet phase, 309
paper
cassette, 293
paper size, correct, 26
parallel port
functionality, 27
path to host memory, 27
parallel port loopback test, 303
parity
selectable, 304
path
correct library, 6
PCMCIA modem card test, 259
`pcmciatest` PCMCIA modem card test, 259
performance
check board, 275
physical
memory, 269
memory test (pmem), 269
physical name, 266
pin connections for loopback plugs, 405
pixel processor test, 333
pltest
command line syntax, 266
controller test
fan module, 264
NVRAM battery module, 264
error messages, 267
options, 264
test modes, 265
`plntest` SPARCstorage array controller test, 263
pmem
command line syntax, 270
error messages, 271
options, 269
UltraSPARC, 270
`pmem` physical memory test, 269

pmem test modes, 271
polygon test
cg6, 56
port
audio, 10
busy, 27
composite video input, 286
SPARCprinter, 25
super VHS, 286
Postscript, 25
Prestoserve test
pstest, 275
print
inages, spdtest, 293
intervals, 293
resolution, 292
print intervals, 27
print warning field
nettest, 254
printer
Centronics, 93
ECP-mode, 93
functional test, 292
test, 25
test pattern, 292
test, hardware and software requirements, 26
printer test
ecpptest subtest, 94
printer tests
`bpptest` SBus printer card test, 25, 231
`lpvitest` SBus printer card test, 25, 231
printing test, 291
printout, test
test
printout, 25
PROMCheck, 281
prompts
default, xxvii
shell, xxvii
protocol tests, 369
pstest
command line syntax, 277
error messages, 277
options, 276
Prestoserve test, 275
test modes, 277
`pstest` prestoserve test, 275

Q

quad ethernet test, 251

R

RAMDAC, overlay registers, 55
random test
nettest, 252
read / write access, 275
realtime
video conferencing, 281
receive timeout field
nettest, 253
remote
testing, 7
requirements
hardware, 6
printer test, 26
software, 6
requirements, hardware
spif test, 301
rtvctest
command line syntax, 286
error messages, 286
options, 284
ports, 285
source format, 285
ports, 285
SunVideo test, 281
verification test modules, 282
`rtvctest` SunVideo test, 281

S

S24 framebuffer test, 369
SBus

DMA circuitry, 25, 94
expansion subsystem, 397
SBus card test, bpptest, 25
SBus expansion subsystem test
xbtest, 397
SBus test
serial ports test sptest, 309
SBus tests
framebuffer test fbtest, 103
serial port controller test, 294, 301
SunVideo test, 281
scalable
nettest, 252
scalable tests, 3
scalable, plntest is not, 265
serial parallel controller test, 294, 301
serial ports test, 309
software
commands, xxvi
software requirements
for syncloop testing, 310
source
video format, 285
space
swap requirements, 391
space swap, 391
SPARCprinter
paper tray magnets, 26
port, 25
SPARCstorage Array (SSA), 263
SPARCstorage array controller test, 263
spdtest
command line syntax, 294
error messages, 295
image, 293
NeWSprinter test, 291
options, 292
print resolution, 292
print server, 291
printer functional test, 292
printer test pattern, 292
test mode, offline, 293
tray, 293
spdtest NeWSprinter test, 291
speakerbox
control buttons, 10
spif
command line syntax, 305
error messages, 306
hardware requirements, 301
intervention mode, 301
options, 302
test mode, offline, 305
spiftest serial parallel controller
test, 294, 301
sptest
command line syntax, 314
error messages, 315
options, 311
test modes, 313
sptest serial ports test, 309
SSA (SPARCstorage Array), 263
SSA controller card, 266
standard
command line arguments, 2
command line arguments, table of, 2
usage, 2
status window
shows test passes, 25
stop bits, 304
stress mode, 3, 4
Sun documents
ordering, xxviii
sunbuttons
command line syntax, 321
error messages, 321
from a .customtest file, 319
from a shell command line, 319
SunButtons Test, 319
test mode, offline, 321
sunbuttons SunButtons graphics
manipulation device test, 319
sundials
command line syntax, 325
error messages, 325
test, 323

test modes, 325
sundials SunDials graphics manipulator test, 323
sunlink HSI/S SBus board test, 327
SunVideo
card, 284
SunVideo test, 281
SunVideo test (rtvctest), 281
SunVTS
accessing, 1
interfaces
CDE, 1
OPEN LOOK, 1
TTY, 1
mode, runs test in, 2
status window, 25
swap space, 391
requirements, 391
reserve option, 391
sxtest
command line syntax, 348
error messages, 348
module descriptions, 336
option menu, 334
pixel processor test, 333
test modes, 347
sxtest Pixel Processor Test, 333
syncloop testing, 309
syntax
audio, command line, 18
cg6, 58
command line, 2
nettest, 255
plntest, 266
pmem, 270
pstest, 277
rtvctest, 286
spdtest, 294
spif, 305
sptest, 314
sunbuttons, 321
sundials, 325
sxtest, 348
systest, 352
tapetest, 361
tcxtest, 374
vmem, 394
xbtest, 400
system
host, 263
load, 3
systest
command line syntax, 352
error messages, 352
options, 351
test modes, 352

T

tapetest
command line syntax, 361
error messages, 363
options, 355
tape drive test, 355
tapetest tape drive test, 355
tapetest test modes, 361
target host field
nettest, 253
target machine(s), 251
TCP/IP (transport control protocol/interface program), 251
tcxtest
AFX protocol tests, 369
colormap tests, 370
command line syntax, 374
cursor tests, 370
error messages, 375
frame buffer memory tests, 369
frame buffer test, 369
has four test groups, 369
options, 373
subtests, 370
test modes, 373
tcxtest S24 framebuffer test, 369
test
audio, 9
bpptest, 25

cdtest, 31
data pattern, bpptest, 27
isdntest, 205
modes
connectivity, 4
offline, 4
online, 4
stress, 4
modes, available, 4
test modes
audio, 17
bpptest, 27
cdtest, 33
cg14test, 49
isdntest, 210
mptest, 244
plntest, 265
pmem, 271
pstest, 277
sptest, 313
sundials, 325
sxtest, 347
systest, 352
tapetest, 361
tcxtest, 373
vmem, 393
xbtest, 400
test modules
rtvctest, 282
testing
remotely, 7
tests
concurrent, 276
token ring test, 251
trace mode, function, 3
transport control protocol/interface program (TCP/IP), 251
tray menu
paper cassette, 293
TTY
interface, 2
keyboard, 304
screen, 304
terminal, 304
txctest
acceleration tests, 369
typographic conventions
conventions
typographic
typeface
symbols, xxvi

U

UltraSPARC
pmem test, 270
UNIX commands, xxvi
usage statement, 2
user tests
`sunbuttons`, 319
`sundials`, 323

V

-v, 2
`valloc`, 391
variable
path name, 6
verbose mode, 2
VHS port, 286
video
composite input port, 286
video conferencing, realtime, 281
video data, digitize, 282
video input port, 285
video source format, 285
virtual memory, 392
vmem test, 391
virtual memory test, 391, 394
vmem
command line syntax, 394
configuration, 392
error messages, 394
options, 392
test modes, 393
`vmem` virtual memory test, 391

W

window
- locking default, 7
- locking disabled, 7
- locking enabled, 7

X

XBox, Sbus expansion subsystem, 397
xbtest
- command line syntax, 400
- error messages, 401
- options, 399
- test modes, 400
- transparent mode test, 397

`xbtest` SBus expansion subsystem test, 397

Z

ZX and TZX graphics accelerator test (leotest), 213