IRIX™ Based Field Diagnostics

Document Number 108-0163-001

Contributors

Written by Darrin Goss
Illustrated by Darrin Goss
Production by Michael Dixon
Engineering contributions by Satish Mirle, Larus Maxwell, and Venkatesh Nayak

IRIX™ Based Field Diagnostics
Document Number 108-0163-001

Silicon Graphics, Inc.
Mountain View, California

Contents

Figures

Tables

About This Guide

This document describes the diagnostic tests that you can use while the IRIX™ operating system is running in an Origin200™, Origin2000™, CRAY® Origin2000, or Onyx2™ system. These diagnostics run from the IRIX prompt and are called IRIX based field diagnostics.

This document is written for System Support Engineers (SSEs) and Field Engineers (FEs) to use as a reference document in training and onsite.

The information that this document contains is organized into the following chapters:

- Chapter 1, "Introduction to the IRIX Based Field Diagnostics," introduces the IRIX based diagnostics, describes how to install them, and describes how to start the field_diag program.
- Chapter 2, "field_diag User Interface," describes the user interface that you can use to run the IRIX based diagnostics.
- Chapter 3, "Diagnostic Tests," describes the individual IRIX based diagnostic tests. It includes instructions on how to run each test from the field_diag user interface and information about the output that each test returns.

This document corresponds to the IRIX based field diagnostics included on the *Internal Support Tools 1.1* CD.

Typographical Conventions

This document uses the following typographical conventions:

- Information displayed on the screen is shown in `Courier` type.
- Text that you enter is shown in **`Courier bold`** type.
- Commands in text, filenames, pathnames, and variables are shown in *italic* type.

Chapter 1

Introduction to the IRIX Based Field Diagnostics

This chapter introduces the IRIX based field diagnostics, describes how to install them, and describes how to start the field_diag program that you can use to run them.

1.1 About the IRIX Based Field Diagnostics

The IRIX based field diagnostics are diagnostic tests that you can run from an IRIX prompt while user operations continue on Origin200, Origin2000, CRAY Origin2000, and Onyx2 systems. There are two components to the IRIX based field diagnostics:

- the field_diag program
- the various diagnostic tests

Note: The System Verification Program (SVP) must be installed to run the field diagnostics because the field diagnostics package uses several of the diagnostics contained in SVP.

1.1.1 About the field_diag Program

The field_diag program is an ASCII user interface that you can use to run any of the IRIX based field diagnostics that are available for the hardware in your system.

Refer to Chapter 2, "field_diag User Interface," for more information about the user interface for the field_diag program.

Note: This interface is identical to the user interface for the field stress tool (FST). The similarity of the two interfaces provides a common "look and feel" for diagnosing hardware failures.

1.1.2 About the Diagnostic Tests

Diagnostic tests are available to test the following hardware components while IRIX is running on a system:

- Ethernet hardware on the BaseIO boards
- floating-point unit on the Node boards
- LINC DMA hardware on the Node boards, ATM boards, and HIPPI-S boards
- MediaIO boards
- memory
- RAID array hardware (restricted to inquiry to the customer-replaceable unit level)
- networking hardware (ATM board, BaseIO board, HIPPI-S board, MENET board)

Refer to Chapter 3, "Diagnostic Tests," for more information about the diagnostic tests that you can run from the field_diag program.

1.2 Installing the IRIX Based Field Diagnostics

The Internal Support Tools group releases the IRIX based field diagnostics on the *Internal Support Tools* CDs. Follow the installation instructions in the CD booklet to install the IRIX based field diagnostics on your system.

Any future updates to the IRIX based field diagnostics will be available from the Internal Support Tools Web site (http://ist.csd.sgi.com/Tools/Home_pages/products.html) or on future CD releases.

This document corresponds to the IRIX based field diagnostics contained on the *Internal Support Tools 1.1* CD.

1.3 Summary of Chapter 1

This chapter introduced the IRIX-based field diagnostics for Origin200, Origin2000, CRAY Origin2000, and Onyx2 systems.

The IRIX based field diagnostics are implemented through two components:

- the field_diag user interface that you can use to load, run, and monitor the diagnostic tests
- the individual diagnostic test programs

Chapter 2

field_diag User Interface

This chapter describes the field_diag user interface.

2.1 About the field_diag User Interface

The field_diag user interface provides a common ASCII interface that you can use to run the IRIX based field diagnostics. This interface enables you to perform the following actions:

- select any available diagnostic test
- modify any user-adjustable parameters for the selected diagnostic test
- run the selected diagnostic the to test the system
- view the output from the selected diagnostic test

This interface is identical to the user interface for the field stress tool (FST). The similarity of the two interfaces provides a common "look and feel" for diagnosing hardware failures.

2.2 Starting the field_diag User Interface

You must have root privilege to run the field_diag user interface.

Perform the following procedure to start the field_diag user interface:

1. Enter the following command to change to the */usr/diags/diags_interface/* directory:

   ```
   cd /usr/diags/diags_interface
   ```

2. Enter the following command to start the field_diag program:

   ```
   ./field_diag
   ```

2.3 field_diag User Interface Components

Figure 2-1 shows the components of the main screen of the field_diag user interface. The text following the figure describes the components.

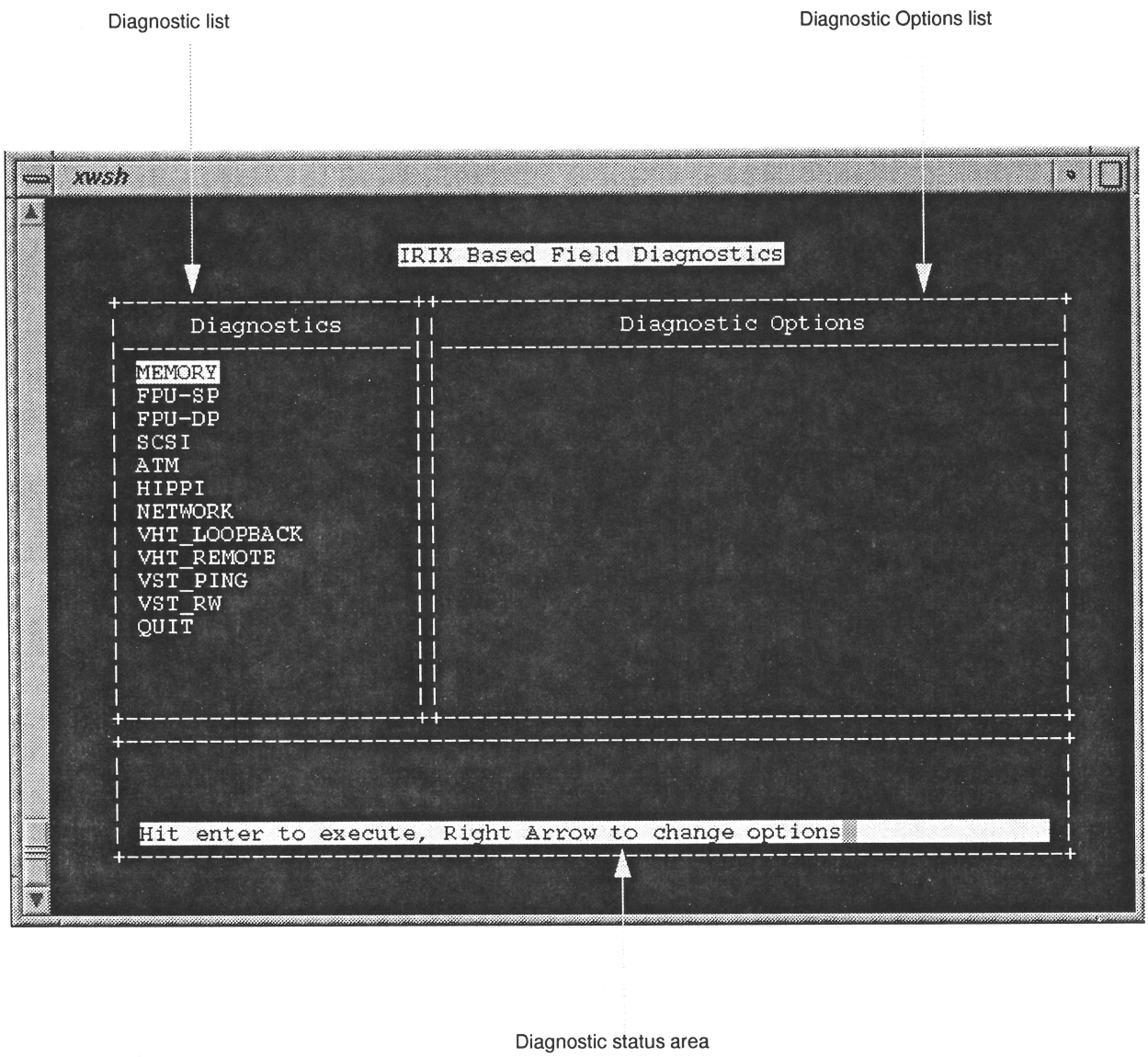

Figure 2-1 field_diag Interface (Main Screen Components)

The main screen includes the following components:

- Diagnostic list
- Diagnostic Options list
- diagnostic status area

When you type Ctrl+T on the main screen, the Test Log Output window appears. This window displays the output messages from the test that is currently running. Figure 2-2 shows the Test Log Output window.

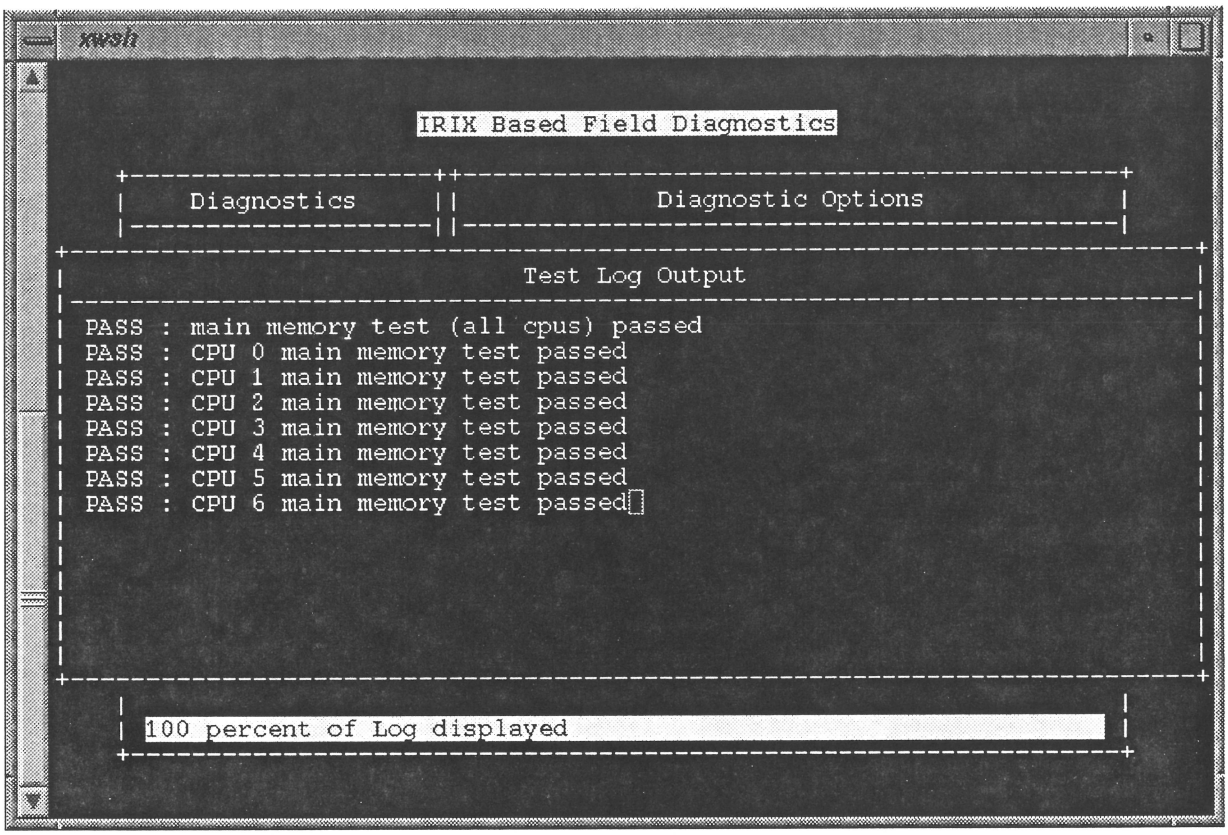

Figure 2-2 field_diag Interface (Test Log Output Window)

2.3.1 Diagnostic List

The Diagnostic list contains the names of all diagnostics that are available for the system configuration that you are using. The field_diag program automatically determines the hardware available in the system being tested and adjusts this list to display only the diagnostics that can test the hardware in the system.

Table 2-1 describes all of the possible options that are available in the Diagnostic list.

Note: The interface that you use onsite displays only the options that correspond to the hardware in your system.

Table 2-1 Possible Diagnostic List Options

Option	Description
ATM	Selects the ATM LINC DMA verification test. Refer to Section 3.1, "ATM LINC DMA Verification Test," for more information.
FPU-SP	Selects the floating-point unit (single-precision) test. Refer to Section 3.2, "Floating-Point Unit (Single-Precision) Test," for more information.
FPU-DP	Selects the floating-point unit (double-precision) test. Refer to Section 3.3, "Floating-Point Unit (Double-Precision) Test," for more information.
HIPPI	Selects the HIPPI-S LINC DMA verification test. Refer to Section 3.4, "HIPPI-S LINC DMA Verification Test," for more information.
MIO	Selects the MediaIO board test. Refer to Section 3.5, "MediaIO Board Test," for more information.
MEMORY	Selects the memory test. Refer to Section 3.6, "Memory Test," for more information.
NETWORK	Selects the network thrasher test. Refer to Section 3.7, "Network Thrasher Test," for more information.
RAID	Selects the RAID verification test. Refer to Section 3.8, "RAID Verification Test," for more information.
SCSI	Selects the SCSI thrasher test. Refer to Section 3.9, "SCSI Thrasher Test," for more information.
VHT_LOOPBACK	Selects the vicious HIPPI test (loopback test). Refer to Section 3.10, "Vicious HIPPI Tests," for more information.
VHT_REMOTE	Selects the vicious HIPPI test (remote test). Refer to Section 3.10, "Vicious HIPPI Tests," for more information.
VST_PING	Selects the vicious socket test (ping test). Refer to Section 3.11, "Vicious Socket Tests," for more information.
VST_RW	Selects the vicious socket test (read and write test). Refer to Section 3.11, "Vicious Socket Tests," for more information.
QUIT	Quits the field_diag program.

Perform any of the following actions to navigate through the Diagnostic list:

- Use the up and down arrow keys to move through the diagnostic tests in the Diagnostic list.
- Use the right arrow key to move from the Diagnostic list into the Diagnostic Options list.
- Use the left arrow key to move from the Diagnostic Options list back into the Diagnostic list.

- Press the Enter key to run the selected test.
- Press the H key to display help messages for the selected test. (Figure 2-3 shows an example of a help message display.)

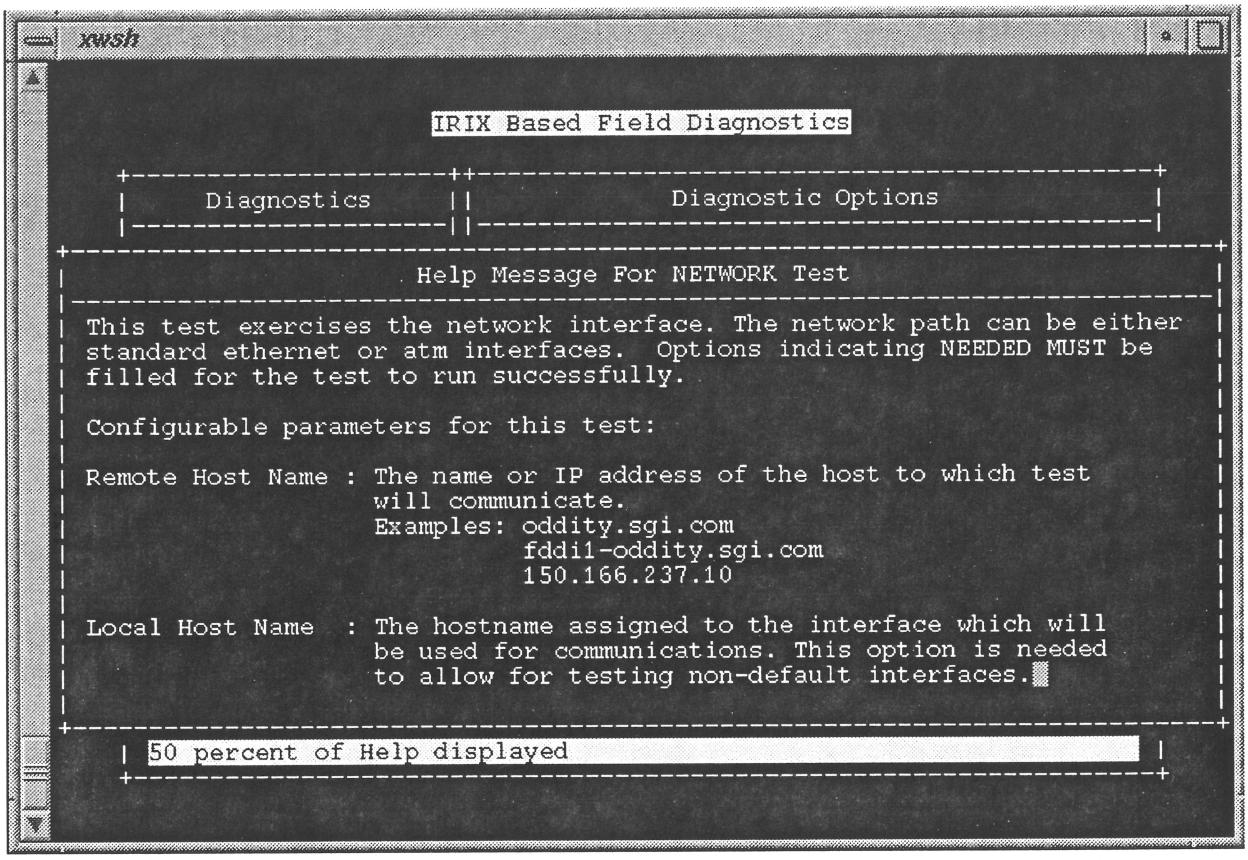

Figure 2-3 Example of Help Message Display (Network Thrasher Test)

2.3.2 Diagnostic Options List

The Diagnostic Options list shows the parameters that you can adjust for each diagnostic test. If nothing appears in this list, the test does not have user-adjustable parameters.

For example, Figure 2-4 shows that the SCSI thrasher test has two parameters that you can adjust: the number of writes that the test performs and the size of the file that the test writes.

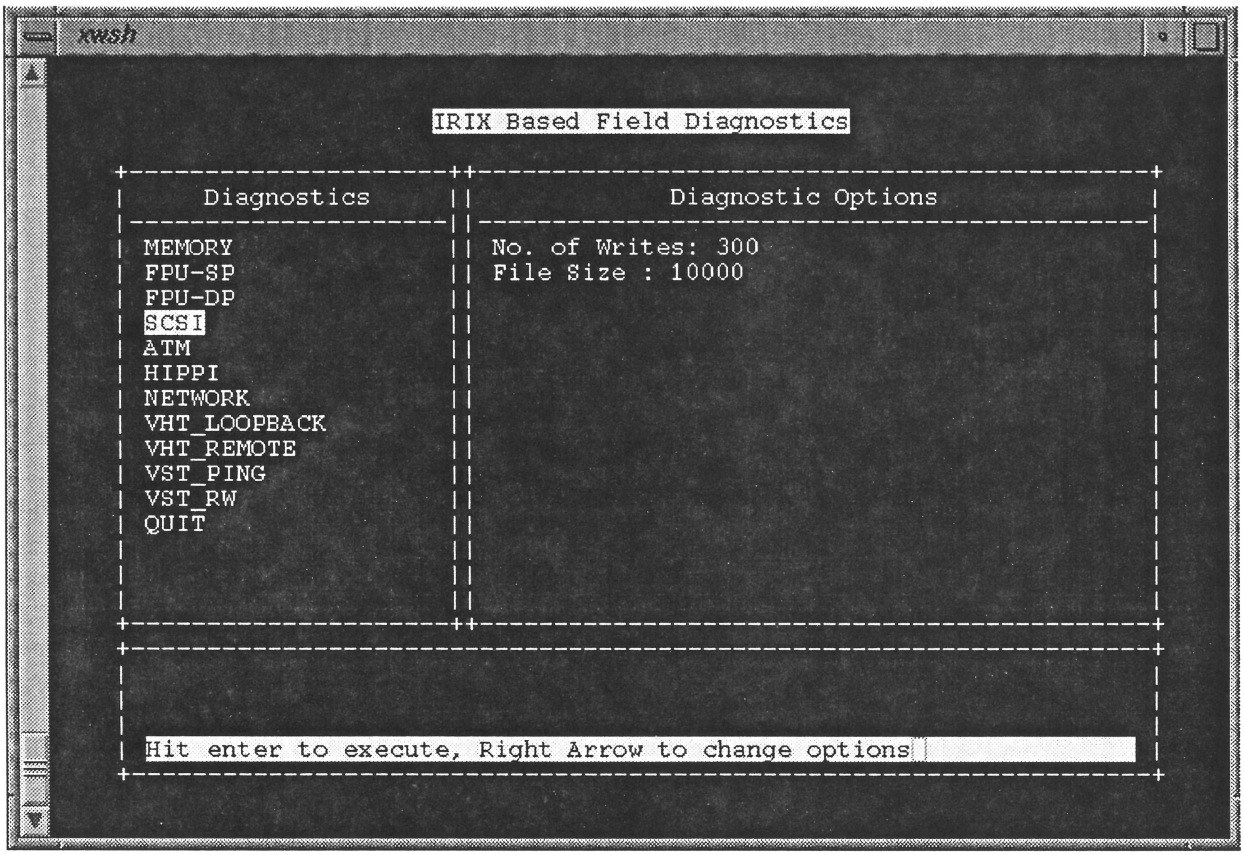

Figure 2-4 Example of Diagnostic Options List (SCSI Thrasher Test)

Perform any of the following actions to navigate through the Diagnostic Options list:

- Use the up and down arrow keys to move through the parameters in the Diagnostic Options list.
- Press the Enter key to select the parameter that you want to change. (When the prompt `Enter New Value for the Option` appears in the diagnostic status area, type the new value, and press the Enter key.)
- Use the left arrow key to move from the Diagnostic Options list into the Diagnostic list.

2.3.3 Diagnostic Status Area

The diagnostic status area provides information about the current status of the interface and diagnostic test that is running. It displays the following information:

- a prompt that indicates what you can do on the current screen (for example, `Hit enter to execute, Right Arrow to change options`)
- a status line that indicates what action the field_diag program is currently performing (for example, `Loading Diags ....` and `Executing : memory`)
- status information about the state of the selected test (for example, `memory Test Passed`)

2.3.4 Test Log Output Window

The Test Log Output window displays output from the test that is currently running. This information includes any error messages that the test generates. If the output contains more than one screen of text, use the up and down arrow keys to scroll through the text. (The diagnostic status area on the main screen displays the percentage of the output that you have viewed.)

Enter Ctrl+T to toggle the display of this window. Refer again to Figure 2-2 for an example of a Test Log Output window.

2.4 Summary of Chapter 2

This chapter described the field_diag user interface.

The field_diag user interface provides a common ASCII interface from which you can run and monitor all IRIX based field diagnostics that are available for the hardware in your system. The interface has four main components:

- the Diagnostic list on the main screen
- the Diagnostic Options list on the main screen
- the diagnostic status area on the main screen
- the Test Output Log window

You can use the interface to perform the following actions:

- to select any available diagnostic test
- to modify any user-adjustable parameters for the selected diagnostic test
- to run the selected diagnostic to test the system
- to view the output from the diagnostic test

Table 2-2 summarizes the keyboard characters that you can use to navigate through the interface.

Table 2-2 Summary of the field_diag Keyboard Commands

Interface Location	Keyboard Character	Navigation
Diagnostic list	Up arrow	Move up to the previous diagnostic test in the list
	Down arrow	Move down to the next diagnostic test in the list
	Right arrow	Move into the Diagnostic Options list
	Enter	Execute the selected diagnostic test
	H	Display the help information for the current test
Diagnostic Options list	Up arrow	Move up to the previous option in the list
	Down arrow	Move down to the next option in the list
	Left arrow	Move into the Diagnostic list
	Enter	Select the parameter to modify
Anywhere	Ctrl+T	Toggle display of the Test Log Output window

Chapter 3

Diagnostic Tests

This chapter describes the following diagnostic tests, which you can run from the field_diag user interface:

- ATM LINC DMA verification test
- floating-point unit (single-precision) test
- floating-point unit (double-precision) test
- HIPPI-S LINC DMA verification test
- MediaIO board test
- memory test
- network thrasher test
- RAID verification test
- SCSI thrasher test
- vicious HIPPI tests
- vicious socket tests

3.1 ATM LINC DMA Verification Test

The ATM LINC DMA verification test verifies that DMA operations between the CPUs and ATM boards are successful. This test uses the following algorithm:

1. The test performs DMA transfers of 8 KB data blocks from the host to LINC0 on the specified ATM network card.
2. The test uses a DMA operation to read the data back into a buffer in memory.
3. The test uses programmed I/O (PIO) operations to access the data in memory and verifies that the data read back with a DMA operation is the same as the data that was written with a DMA operation.
4. The test repeats Steps 1 through 3 using LINC1 on the network card instead of LINC0.

If this test fails, the failing field-replaceable unit (FRU) is a Node board or ATM board.

3.1.1 Prerequisites for Running the ATM LINC DMA Verification Test

This test has the following prerequisites:

- The system that you want to test must have the IRIX operating system booted and running.
- You must run this test from superuser mode.

3.1.2 Running the ATM LINC DMA Verification Test

To run the ATM LINC DMA verification test from the field_diag program, perform the following procedure:

1. Select the ATM option from the Diagnostic list.
2. Press the `Enter` key to start running the ATM LINC DMA verification test.

   The message `Executing : ATM` appears in the diagnostic status area, indicating that the test is running.
3. View the diagnostic status area and Test Log Output window for messages from the ATM LINC DMA verification test.

Note: The System Verification Program (SVP) also runs this test. For more information about SVP, refer to the *System Verification Program Reference Guide*, part number 108-0165-002.

3.1.3 Output From the ATM LINC DMA Verification Test

This test returns output to the diagnostic status area of the user interface and the Test Log Output window.

3.1.3.1 Pass Output

If this test completes without detecting any hardware failures, the diagnostic status area of the user interface displays the message `ATM TEST Passed`.

3.1.3.2 Failure Output

If this test detects a hardware failure or encounters an error that makes it unable to continue testing, the diagnostic status area of the user interface displays the message `ATM Test Failed`.

If this message appears, view the Test Log Output window for more information about the cause of the failure.

3.2 Floating-Point Unit (Single-Precision) Test

The floating-point unit (single-precision) test uses matrix manipulations of single-precision numbers to test the floating-point unit. If this test fails, the failing FRU is the Node board that contains the floating-point unit being tested.

3.2.1 Prerequisites for Running the Floating-Point Unit (Single-Precision) Test

This test has the following prerequisites:

- The system that you are testing must have the IRIX operating system booted and running.
- The FORTRAN libraries must be located in */usr/lib/libftn.so*.

3.2.2 Running the Floating-Point Unit (Single-Precision) Test

To run the floating-point unit (single-precision) test from the field_diag program, perform the following procedure:

1. Select the FPU-SP option from the Diagnostic list.
2. If you want to modify the parameter `Number of Iterations` (which specifies the number of matrix manipulations that the test should perform), perform the following actions; otherwise, skip to Step 4.
   - Press the right arrow key to move into the Diagnostic Options list to select the `Number of Iterations` parameter.
   - Press the **`Enter`** key.
   - At the prompt `Enter New Value for the Option` in the diagnostic status area, enter the new value.
   - Press the **`Enter`** key.
3. Press the left arrow key to return to the Diagnostic list.
4. Press the **`Enter`** key to start running the floating-point unit (single-precision) test.

   The message `Executing : Floating point single precision` appears in the diagnostic status area, indicating that the test is running.
5. View the diagnostic status area and Test Log Output window for messages from the floating-point unit (single-precision) test.

Note: The System Verification Program (SVP) also runs this test. For more information about SVP, refer to the *System Verification Program Reference Guide*, part number 108-0165-002.

3.2.3 Output From the Floating-Point Unit (Single-Precision) Test

This test returns output to the diagnostic status area of the user interface and the Test Log Output window.

3.2.3.1 Pass Output

If this test completes without detecting any hardware failures, the diagnostic status area of the user interface displays the message `Floating point single precision Test Passed`.

3.2.3.2 Failure Output

If this test detects a hardware failure or encounters an error that makes it unable to continue testing, the diagnostic status area of the user interface displays the message `Floating point single precision Test Failed`.

If this message appears, view the Test Log Output window for more information about the cause of the failure.

3.3 Floating-Point Unit (Double-Precision) Test

The floating-point unit (double-precision) test uses matrix manipulations of double-precision numbers to test the floating-point unit. If this test fails, the failing FRU is the Node board that contains the floating-point unit that is being tested.

3.3.1 Prerequisites for Running the Floating-Point Unit (Double-Precision) Test

This test has the following prerequisites:

- The system that you are testing must have the IRIX operating system booted and running.
- The FORTRAN libraries must be located in */usr/lib/libftn.so*.

3.3.2 Running the Floating-Point Unit (Double-Precision) Test

To run the floating-point unit (double-precision) test from the field_diag program, perform the following procedure:

1. Select the FPU-DP option from the Diagnostic list.
2. If you want to modify the parameter `Number of Iterations` (which specifies the number of matrix manipulations that the test should perform), perform the following actions; otherwise, skip to Step 4.
   - Press the right arrow key to move into the Diagnostic Options list to select the parameter `Number of Iterations`.
   - Press the **`Enter`** key.
   - At the prompt `Enter New Value for the Option` in the diagnostic status area, enter the new value.
   - Press the **`Enter`** key.
3. Press the left arrow key to return to the Diagnostic list.
4. Press the **`Enter`** key to start running the floating-point unit (double-precision) test.

   The message `Executing : Floating point double precision` appears in the diagnostic status area, indicating that the test is running.
5. View the diagnostic status area and Test Log Output window for messages from the floating-point unit (double-precision) test.

Note: The System Verification Program (SVP) also runs this test. For more information about SVP, refer to the *System Verification Program Reference Guide*, part number 108-0165-002.

3.3.3 Output From the Floating-Point Unit (Double-Precision) Test

This test returns output to the diagnostic status area of the user interface and the Test Log Output window.

3.3.3.1 Pass Output

If this test completes without detecting any hardware failures, the diagnostic status area of the user interface displays the message `Floating point double precision Test Passed`.

3.3.3.2 Failure Output

If this test detects a hardware failure or encounters an error that makes it unable to continue testing, the diagnostic status area of the user interface displays the message `Floating point double precision Test Failed`.

If this message appears, view the Test Log Output window for more information about the cause of the failure.

3.4 HIPPI-S LINC DMA Verification Test

The HIPPI-S LINC DMA verification test verifies that DMA operations between the CPUs and HIPPI-S boards are successful. This test uses the following algorithm:

1. The test performs DMA transfers of 8 KB data blocks from the host to LINC0 on the specified HIPPI-S network card.
2. The test uses a DMA operation to read the data back into a buffer in memory.
3. The test uses programmed I/O (PIO) operations to access the data in memory and verifies that the data read back with a DMA operation is the same as the data that was written with a DMA operation.
4. The test repeats Steps 1 through 3 using LINC1 on the network card instead of LINC0.

If this test fails, the failing FRU is a Node board or HIPPI-S board.

3.4.1 Prerequisites for Running the HIPPI-S LINC DMA Verification Test

This test has the following prerequisites:

- The system that you want to test must have the IRIX operating system booted and running.
- You must run this test from superuser mode.

3.4.2 Running the HIPPI-S LINC DMA Verification Test

To run the HIPPI-S LINC DMA verification test from the field_diag program, perform the following procedure:

1. Select the HIPPI option from the Diagnostic list.
2. Press the **Enter** key to start running the LINC DMA verification test.

   The message `Executing : HIPPI` appears in the diagnostic status area, indicating that the test is running.
3. View the diagnostic status area and Test Log Output window for messages from the HIPPI-S LINC DMA verification test.

Note: The System Verification Program (SVP) also runs this test. For more information about SVP, refer to the *System Verification Program Reference Guide*, part number 108-0165-002.

3.4.3 Output From the HIPPI-S LINC DMA Verification Test

This test returns output to the diagnostic status area of the user interface and the Test Log Output window.

3.4.3.1 Pass Output

If this test completes without detecting any hardware failures, the diagnostic status area of the user interface displays the message `HIPPI TEST Passed`.

3.4.3.2 Failure Output

If this test detects a hardware failure or encounters an error that makes it unable to continue testing, the diagnostic status area of the user interface displays the message `HIPPI Test Failed`.

If this message appears, view the Test Log Output window for more information about the cause of the failure.

3.5 MediaIO Board Test

The MediaIO (MIO) board test checks any MIO boards in the system. It tests the keyboard and mouse ports, audio hardware (headphone port, microphone point, line-out jack, line-in jack, ADAT connector, AES connector, RAD chip, parallel port, serial port, and UST clock signal), and AES Out signal. If this test fails, the failing FRU is the MediaIO board.

3.5.1 Prerequisites for Running the MediaIO Board Test

This test has the following prerequisites:

- The system that you want to test must have the IRIX operating system booted and running.
- You must connect a printer to the parallel port to run this test. If you do not connect a printer, the test fails.
- You must connect a keyboard and mouse to the keyboard/mouse port to run this test. If you do not connect a keyboard and mouse, the test fails.

3.5.2 Running the MediaIO Board Test

To run the MediaIO board test from the field_diag program, perform the following procedure:

1. Select the MIO option from the Diagnostic list.
2. Press the `Enter` key to start running the MediaIO board test.

   The message `Executing : MIO Test` appears in the diagnostic status area, indicating that the test is running.
3. View the diagnostic status area and Test Log Output window for messages from the MediaIO board test.

Note: The System Verification Program (SVP) also runs this test. For more information about SVP, refer to the *System Verification Program Reference Guide*, part number 108-0165-002.

3.5.3 Output From the MediaIO Board Test

This test returns output to the diagnostic status area of the user interface and the Test Log Output window.

3.5.3.1 Pass Output

If this test completes without detecting any hardware failures, the diagnostic status area of the user interface displays the message `MIO Test PASSED`.

3.5.3.2 Failure Output

If this test detects a hardware failure or encounters an error that makes it unable to continue testing, the diagnostic status area of the user interface displays the message `MIO Test Failed`.

If this message appears, view the Test Log Output window for more information about the cause of the failure. Possible messages include:

- `Digital Test on RAD1.AESOut -> RAD1.AESIn: FAIL`

  The test failed. The failing FRU is the MIO board being tested.

- `rad_diag Status: FAIL`

  The test failed. The failing FRU is the MIO board being tested.

- ```
  rad_diag: RSLT:FAIL bad return code
  comparison failed
  loopback AES failed
  comparison failed
  loopback AES failed
  comparison failed
  loopback AES failed
  comparison failed
  loopback AES failed
  ```

  The test failed. The failing FRU is the MIO board being tested.
  ```

3.6 Memory Test

The memory test allocates a portion of memory and then performs bit-pattern testing on that memory. The amount of memory that this test uses depends on the limits that are set by the operating system This test attempts to reserve 85 percent of the available memory for testing.

If this test fails, the failing FRU is a DIMM on the Node that the test specifies.

3.6.1 Prerequisites for Running the Memory Test

This test has the following prerequisites:

- The system that you want to test must have the IRIX operating system booted and running.
- You must run this test from superuser mode.

3.6.2 Running the Memory Test

To run the memory test from the field_diag program, perform the following procedure:

1. Select the MEMORY option from the Diagnostic list.
2. Press the `Enter` key to start running the memory test.

   The message `Executing : memory` appears in the diagnostic status area, indicating that the test is running.
3. View the diagnostic status area and Test Log Output window for messages from the memory test.

Note: The System Verification Program (SVP) also runs this test. For more information about SVP, refer to the *System Verification Program Reference Guide*, part number 108-0165-002.

3.6.3 Output From the Memory Test

This test returns output to the diagnostic status area of the user interface and the Test Log Output window.

3.6.3.1 Pass Output

If this test completes without detecting any hardware failures, the diagnostic status area of the user interface displays the message `memory Test Passed`.

3.6.3.2 Failure Output

If this test detects a hardware failure or encounters an error that makes it unable to continue testing, the diagnostic status area of the user interface displays the message `memory Test Failed`.

If this message appears, view the Test Log Output window for more information about the cause of the failure.

3.7 Network Thrasher Test

The network thrasher test stresses the specified network hardware by sending large blocks of data to a remote system and then reading the data back.

The test uses the following client/server model:

- Multiple clients on the system that you are testing send data to a server on a remote system through a network connection.
- The remote server returns the data to a server on the system that you are testing.

The test compares the data that was returned with the data that was sent. (Figure 3-1 shows an example of this process with three clients.)

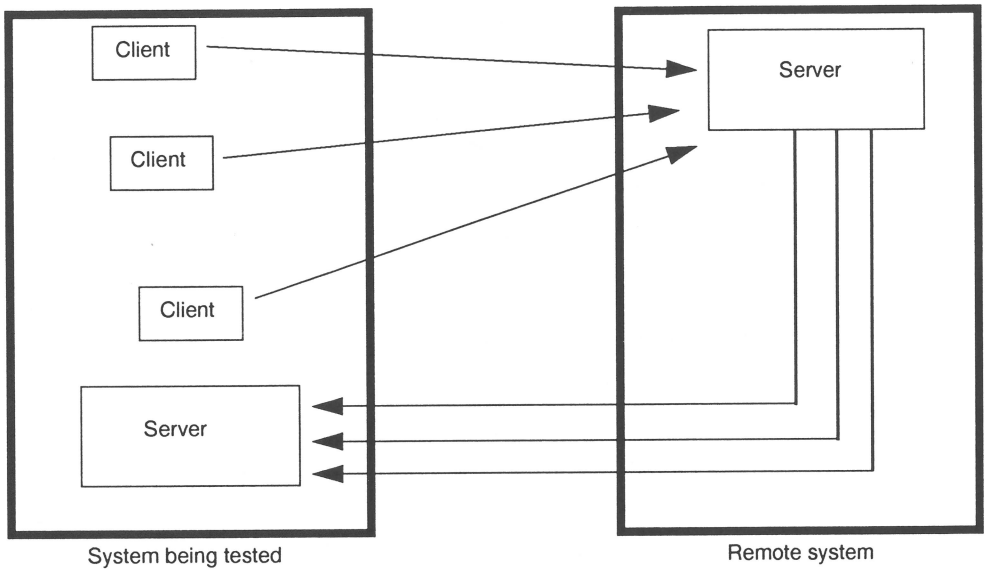

Figure 3-1 Network Thrasher Test Flow

The client processes run on the system that you are testing and simultaneously perform the following actions:

- Each client sends the address to which the server on the remote system should return the data.
- Each client transfers a large file (*/unix*) to the server on the remote system.

One server process runs on the remote system and forks a process for each connection that it receives from a client. Each forked process performs the following actions:

- The process reads the data sent from the client on the system being tested.
- The process returns the data to a server process on the system being tested.

The server process on the system that you are testing receives the data from the server on the remote system. The test compares the data that was received with the data that was sent.

If this test fails, the failing FRU is the BaseIO board

3.7.1 Prerequisites for Running the Network Thrasher Test

This test has the following prerequisites:

- The system that you are testing must have the IRIX operating system booted and running.
- You must have access to a remote system that is functional and running the IRIX operating system.
- You must specify values for all parameters in the Test Options list that are set to `NEEDED` by default.
- You must have disk space available on the system that you are testing to create the temporary files that this test uses. The amount of disk space that this test needs depends on the number of clients that you specify it should fork.
- You must set the `remote host` parameter correctly for the system that you are testing.
- The */etc/hosts* file must have the proper information (IP address with fully qualified hostname) for the system that you want to test. For example:

  ```
  150.166.14.31 bootleg.csd.sgi.com bootleg
  ```

- The remote system must allow login as the guest account.

3.7.2 Running the Network Thrasher Test

To run the network thrasher test from the field_diag program, perform the following procedure:

1. Select the NETWORK option from the Diagnostic list.
2. If you want to modify any of the parameters, perform the following actions; otherwise, skip to Step 4.
   - Press the right arrow key to move into the Diagnostic Options list.
   - Select the parameter that you want to modify. (Table 3-1 describes the parameters that you can modify.)
   - Press the **`Enter`** key.
   - At the prompt `Enter New Value for the Option` in the diagnostic status area, enter the new value.
   - Press the **`Enter`** key.

Table 3-1 Network Thrasher User-Adjustable Parameters

Parameter	Description
`Remote Host Name`	Specifies the remote host to which the local host will communicate. (Enter a hostname or IP address; for example, oddity.sgi.com or 150.166.237.10.) You must set this parameter.
`Number of Threads`	Specifies the number of threads (clients) that the test should use. This test causes extreme stress on the network. Use caution when you adjust the number of threads that this test will use. Setting this parameter too high with adversely affect network performance and the performance of other systems on the network.
`Local Network Path`	Specifies the hostname that is assigned to the interface to be used for communications. (This option enables you to test nondefault interfaces. Enter a hostname; for example, bootleg.sgi.com or atm-bootleg.sgi.com) You must set this parameter.

3. Press the left arrow key to return to the Diagnostic list.
4. Press the **`Enter`** key to start running the network thrasher test.

   The message `Executing : network` appears in the diagnostic status area, indicating that the test is running.
5. View the diagnostic status area and Test Log Output window for messages from the network thrasher test.

Note: The System Verification Program (SVP) also runs this test. For more information about SVP, refer to the *System Verification Program Reference Guide*, part number 108-0165-002.

3.7.3 Output From the Network Thrasher Test

This test returns output to the diagnostic status area of the user interface and the Test Log Output window.

3.7.3.1 Pass Output

If this test completes without detecting any hardware failures, the diagnostic status area of the user interface displays the message `network Test Passed`.

3.7.3.2 Failure Output

If this test detects a hardware failure or encounters an error that makes it unable to continue testing, the diagnostic status area of the user interface displays the message `network Test Failed`.

If this message appears, view the Test Log Output window for more information about the cause of the failure. Possible messages include:

- `ERROR: Network Thrasher Failed. Check error log for error`

  The test failed with an error. This error occurs when there is a data integrity error after the data transfer is completed. The failing FRU is the board that contains the network interface being tested.

- `ERROR: Remote System is not responding. Cannot Continue`

  The test failed with an error. This error occurs when the test checks to determine whether the remote system is accessible. The remote system may not be responding for many reasons. (For example, the network is not functioning, the remote system is down, or the test system is not configured correctly.)

- `ERROR: Could not fork off requested number of clients`

  The test failed with an error. This error occurs if the number of clients is greater than the number of processes that can be forked off.

3.8 RAID Verification Test

The RAID verification test creates an inventory of all customer-replaceable units in an attached RAID array. This test does not write to or read from the RAID array. However, it does query the disks with pass-through commands, which checks the integrity of the SCSI bus to the disk drive level.

This test writes the configuration information into */usr/diags/raid/configfile-x-y* files, where *x* specifies the fiber channel controller number and *y* specifies the RAID unit number. (The test creates multiple files for the RAID array.)

You should run this test when the RAID array is first installed (after initial power-on). After the test passes, run the System Verification Program (SVP) to send the configuration to the MOSAIC database.

If this test fails, the failing FRU is a disk drive or RAID controller.

3.8.1 Prerequisite for Running the RAID Verification Test

This test has the following prerequisite: the system that you want to test must have the IRIX operating system booted and running.

3.8.2 Running the RAID Verification Test

1. Select the RAID option from the Diagnostic list.
2. Press the `Enter` key to start running the RAID verification test.

   The message `Executing : RAID` appears in the diagnostic status area, indicating that the test is running.
3. View the diagnostic status area and Test Log Output window for messages from the RAID verification test.

Note: The System Verification Program (SVP) also runs this test. For more information about SVP, refer to the *System Verification Program Reference Guide*, part number 108-0165-002.

3.8.3 Output From the RAID Verification Test

This test returns output to the diagnostic status area of the user interface and the Test Log Output window.

3.8.3.1 Pass Output

If this test completes without detecting any hardware failures, the diagnostic status area of the user interface displays the message `RAID Passed`.

3.8.3.2 Failure Output

If this test detects a hardware failure or encounters an error that makes it unable to continue testing, the diagnostic status area of the user interface displays the message `RAID Failed`.

If this message appears, view the Test Log Output window for more information about the cause of the failure.

3.9 SCSI Thrasher Test

The SCSI thrasher test stresses the SCSI subsystem. This diagnostic tests the SCSI hardware on MSCSI and BaseIO boards.

The test creates multiple processes that perform synchronized writes and reads. Each process writes a data pattern to a SCSI drive and then waits for the other processes to finish writing data patterns. Once all processes have written their data patterns, the test reads the data back and compares the data that was read back with the data that was written. This test uses five different data patterns.

This test uses the following testing algorithm:

1. The tests mounts any SCSI drives and starts a semaphore program. (The semaphore program synchronizes all disk processes to ensure that no read operations are performed before all processes have completed writing data.)
2. The test creates a */tmp* directory on the system being tested if the directory does not already exist.
3. The test starts two copies of a *diskthrash* process for each mounted disk.
4. Each of the *diskthrash* processes writes data to two files in the */tmp* directory. The data files contain random numbers.
5. All processes wait for the semaphore program to signal that all processes have finished writing data to the */tmp* directory.
6. All processes read data back from the SCSI drives at the same time, which stresses the I/O channels with maximum traffic.
7. The test compares the data that was read back with the data that was written.
8. The test deletes all temporary files it created, deletes all temporary directories that it created, and unmounts any SCSI drives that it mounted.
9. The test generates a pass or failure message and stops.

Note: The *diskthrash* processes exit if they detect more than ten errors during the write process or read process.

If this test fails, the failing FRU is the BaseIO board, MSCSI board, or disk drive that it is testing.

3.9.1 Prerequisites for Running the SCSI Thrasher Test

This test has the following prerequisites:

- The system that you want to test must have the IRIX operating system booted and running.
- You must have disk space available on the system that you are testing to create the temporary files that this test uses.

3.9.2 Running the SCSI Thrasher Test

Caution: Run only one copy of this test at a time. If you run more than one copy at a time, the system may hang.

1. Select the SCSI option from the Diagnostic list.
2. If you want to modify any of the parameters, perform the following actions; otherwise, skip to Step 4.
   - Press the right arrow key to move into the Diagnostic Options list.
   - Select the parameter that you want to modify. (Table 3-2 describes the parameters that you can modify.)
   - Press the **Enter** key.
   - At the prompt `Enter New Value for the Option` in the diagnostic status area, enter the new value.
   - Press the **Enter** key.

Table 3-2 SCSI Thrasher Test User-Adjustable Parameters

Parameter	Description
`No. of Writes`	Specifies the number of writes that the test should perform.
`File Size`	Specifies the size of the files that the test should generate.

3. Press the left arrow key to return to the Diagnostic list.
4. Press the **Enter** key to start running the SCSI thrasher test.

   The message `Executing : randomthrash` appears in the diagnostic status area, indicating that the test is running.
5. View the diagnostic status area and Test Log Output window for messages from the SCSI thrasher test.

Note: The System Verification Program (SVP) also runs this test. For more information about SVP, refer to the *System Verification Program Reference Guide*, part number 108-0165-002.

3.9.3 Output From the SCSI Thrasher Test

This test returns output to the diagnostic status area of the user interface and the Test Log Output window.

3.9.3.1 Pass Output

If this test completes without detecting any hardware failures, the diagnostic status area of the user interface displays the message `randomthrash Passed`.

3.9.3.2 Failure Output

If this test detects a hardware failure or encounters an error that makes it unable to continue testing, the diagnostic status area of the user interface displays the message `randomthrash Failed`.

If this message appears, view the Test Log Output window for more information about the cause of the failure. Possible messages include:

- `ERROR : Diskthrasher Thrasher Failed. Check error.log for error`

  The test failed. The *error.log* file contains more information about the failure. The failing FRU is the BaseIO board, MSCSI board, or disk drive.

- `ERROR : Data ERROR on device` *device_name*
  `Data mismatched at disk offset` *value* `Expected data` *expected* `Act.` *actual*

  The test failed. The data that was read back did not equal the data that was written. The failing FRU is the BaseIO board, MSCSI board, or disk drive.

3.10 Vicious HIPPI Tests

The vicious HIPPI tests are exercisers that test HIPPI channel pairs. There are two vicious HIPPI tests:

- The VHT_LOOPBACK test uses raw channel HIPPI data to exercise the configured HIPPI network device on the system that you are testing. It sends and receives data through a loopback cable on the HIPPI-S board. It then compares the actual data with expected values.
- The VHT_REMOTE tests uses raw channel HIPPI data to exercise the configured HIPPI network device on the system that you are testing. It can run on one system with a loopback cable installed or between two connected systems.

  To exercise the HIPPI interface on one system, it sends and receives data through the loopback cable on the HIPPI-S board. It then compares the actual data with expected values. To run the test on one system, set the `Local Interface` and `Remote Interface` parameters to the same value. (Refer to Table 3-4.)

  To exercise the HIPPI interface using two systems, it writes data to the remote system and reads data back to the local system. It then compares the actual data with expected values.

If one of these tests fails, the failing hardware may be the HIPPI network interface being tested. The failing FRU may be the HIPPI-S board that contains the failing network interface.

3.10.1 Prerequisites for Running the Vicious HIPPI Tests

These tests have the following prerequisites:

- The system that you want to test must have the IRIX operating system booted and running.
- You must specify values for all parameters in the Test Options list that are set to `NEEDED` by default.
- You must use configure the HIPPI device down with the *ifconfig* command (for example, *ifconfig hip0 down*).
- To run the VHT_LOOPBACK test, you must connect a loopback cable on the HIPPI-S board that you want to test.
- To run the VHT_REMOTE test on one system, you must connect a loopback cable on the HIPPI-S board that you want to test.
- To run the VHT_REMOTE test on two systems, you must have access to a functional remote system that is running the IRIX operating system and has the same version of the vicious HIPPI tests installed.

3.10.2 Running the Vicious HIPPI Tests

You can run the vicious HIPPI tests from the field_diag program.

3.10.2.1 Running the VHT_LOOPBACK Test

To run the VHT_LOOPBACK test from the field_diag program, perform the following procedure:

1. Select the VHT_LOOPBACK option from the Diagnostic list.
2. If you want to modify any of the parameters, perform the following actions; otherwise, skip to Step 4.
   - Press the right arrow key to move into the Diagnostic Options list.
   - Select the parameter that you want to modify. (Table 3-3 describes the parameters that you can modify.)
   - Press the **Enter** key.
   - At the prompt `Enter New Value for the Option` in the diagnostic status area, enter the new value.
   - Press the **Enter** key.

Table 3-3 VHT_LOOPBACK Test User-Adjustable Parameters

Parameter	Description
`num. packets`	Specifies the number of data packets that the test will send.
`Pattern`	Specifies the data pattern that the test should use. Enter a hexadecimal number or any of the following settings: `bits` specifies that each 64-bit data word will have a random sequence of consecutive 1 bits. `slide0` specifies that the first 64-bit data word will have all bits (except bit 0) set to 1; bit 0 will be set to 0. Subsequent data words will be circularly left-shifted by one bit position. `slide1` specifies that the first 64-bit data word will have all bits (except bit 0) set to 0; bit 0 will be set to 1. Subsequent data words will be circularly left-shifted by one bit position. `random` specifies that data words will be set to random values. `all` specifies that the test should use built-in data patterns, one per pass, in a circular pattern.
`Block size`	Specifies the block size to use for I/O operations (in bytes). Enter `pass` or one or more values in the following format: *max* [:*min*[:*step*]] When you use the *min* value, the test randomly generates a block size value in the range from *min* to *max*. If you specify a *step* value, the randomly selected value increases by the value of *step* for each pass of the test, until the test reaches the *max* value. When you use the `pass` option, the test sets the blocksize to the current pass count value.

Table 3-3 (continued) VHT_LOOPBACK Test User-Adjustable Parameters

Parameter	Description
`Interface`	Specifies the device name of the HIPPI interface that the test should use. (For example, /dev/hippi1)

3. Press the left arrow key to return to the Diagnostic list.
4. Press the **`Enter`** key to start running the VHT_LOOPBACK test.

   The message `Executing : VHT_LOOPBACK` appears in the diagnostic status area, indicating that the test is running.
5. View the diagnostic status area and Test Log Output window for messages from the VHT_LOOPBACK test.

3.10.2.2 Running the VHT_REMOTE Test

To run the VHT_REMOTE test from the field_diag program, perform the following procedure:

1. Select the VHT_REMOTE option from the Diagnostic list.
2. If you want to modify any of the parameters, perform the following actions; otherwise, skip to Step 4.
   - Press the right arrow key to move into the Diagnostic Options list.
   - Select the parameter that you want to modify. (Table 3-4 describes the parameters that you can modify.)
   - Press the **Enter** key.
   - At the prompt `Enter New Value for the Option` in the diagnostic status area, enter the new value.
   - Press the **Enter** key.

Table 3-4 VHT_REMOTE Test User-Adjustable Parameters

Parameter	Description
`num. packets`	Specifies the number of data packets that the test will send.
`remote host`	Specifies the remote system with which the test will communicate. Enter a hostname or IP address. You must set this parameter.
`Pattern`	Specifies the data pattern that the test should use. Specify a hexadecimal number or any of the following settings: `bits` specifies that each 64-bit data word will have a random sequence of consecutive 1 bits. `slide0` specifies that the first 64-bit data word will have all bits (except bit 0) set to 1; bit 0 will be set to 0. Subsequent data words will be circularly left-shifted by one bit position. `slide1` specifies that the first 64-bit data word will have all bits (except bit 0) set to 0; bit 0 will be set to 1. Subsequent data words will be circularly left-shifted by one bit position. `random` specifies that data words will be set to random values. `all` specifies that the test should use built-in data patterns, one per pass, in a circular pattern.
`Block size`	Specifies the block size to use for I/O operations (in bytes). Specify `pass` or one or more values in the following format: *max* [:*min*[:*step*]] When you use the *min* value, the test randomly generates a block size value in the range from *min* to *max*. If you specify a *step* value, the randomly selected value increases by the value of *step* for each pass of the test, until the test reaches the *max* value. When you use the `pass` option, the test sets the blocksize to the current pass count value.

Table 3-4 (continued) VHT_REMOTE Test User-Adjustable Parameters

Parameter	Description
`I-field`	Specifies the I-field value. See the */usr/etc/hippi* file or use the *hipmap* command to determine the I-field value. (For example, 0x0300001d, 0x01000004, and 0x3c) You must set this parameter. If you are using one system in loopback mode or two systems that are directly connected (that is, no HIPPI switch is used), the value of this parameter is not important. Set this parameter to 0x00000000 in either of these situations.
`Local Interface`	Specifies the device name of the HIPPI interface on the local system. (This device will listen for writes. For example, /dev/hippi0) You must set this parameter.
`Remote Interface`	Specifies the device name of the HIPPI interface on the remote system. (This device will send the writes. For example, /dev/hippi1) To run the test in loopback mode, set the `Local Interface` and `Remote Interface` parameters to the same value. You must set this parameter.

3. Press the left arrow key to return to the Diagnostic list.
4. Press the **`Enter`** key to start running the VHT_REMOTE test.

   The message `Executing : VHT_REMOTE` appears in the diagnostic status area, indicating that the test is running.
5. View the diagnostic status area and Test Log Output window for messages from the VHT_REMOTE test.

3.10.3 Output From the Vicious HIPPI Tests

This test returns output to the diagnostic status area of the user interface and the Test Log Output window.

3.10.3.1 Pass Output

If the VHT_LOOPBACK test completes without detecting any hardware failures, the diagnostic status area of the user interface displays the message `VHT_LOOPBACK Test Passed`.

If the VHT_REMOTE test completes without detecting any hardware failures, the diagnostic status area of the user interface displays the message `VHT_REMOTE Test Passed`.

3.10.3.2 Failure Output

If the VHT_LOOPBACK test detects a hardware failure or encounters an error that makes it unable to continue testing, the diagnostic status area of the user interface displays the message `VHT_LOOPBACK Test Failed`.

If the VHT_REMOTE test detects a hardware failure or encounters an error that makes it unable to continue testing, the diagnostic status area of the user interface displays the message `VHT_REMOTE Test Failed`.

If one of these messages appears, view the Test Log Output window for more information about the cause of the failure. Possible messages include:

- `***** Data Compare Error *****`
  `Differences written to "`*file*`".`
  `Current pass = 1 Current data pattern = slide1`

  The test failed because the actual data did not equal the expected data. The failing FRU may be the HIPPI-S board. Refer to the generated error file for more information.

  The following output shows a partial sample of the contents of an error file:

```
Pass : 1
A : expected
B : actual
x : logical difference

A( 0)+ 8000000000000198 0000000100800080 *................*
B( 0)+ 8000000000000198 0000000000000000 *................*
x( 0)+ 0000000000000000 0000000100800080 *................*
A(10)+ 00000000007f0000 0000000000000006 *................*
B(10)+ 0000000000000001 0000000000000002 *................*
x(10)+ 00000000007f0001 0000000000000004 *................*
A(20)+ 00000000001ffe00 00000fffffffe00 *................*
B(20)+ 0000000000000004 0000000000000008 *................*

x(20)+ 00000000001ffe04 00000fffffffe08 *................*
```

- ```
  Executing Write Test
  connect() failed, errno = 152, text = "Connection refused".
  ```

  A write operation failed because there is no remote system to receive the data that it writes. This failure could occur because the remote system is not running or because the network interface is failing. The failing FRU may be the HIPPI-S board.
  ```

3.11 Vicious Socket Tests

The vicious socket tests are socket-based exercisers that use TCP sockets to send and receive data across a network. There are two vicious socket tests:

- The VST_PING test uses ICMP to send data packets to a remote system, which then returns the packets to the vicious socket test process on the local system. The local system verifies that the data returned from the remote system is the same as the data sent to the remote system. You must run this test as root.
- The VST_RW test reads and writes data patterns between two systems. It compares the data that is sent to the remote system with the data that is received from the remote system.

If one of these tests fails, the failing hardware is the network interface being tested. The failing FRU is the board that contains the failing network interface.

3.11.1 Prerequisites for Running the Vicious Socket Tests

These tests have the following prerequisites:

- The system that you want to test must have the IRIX operating system booted and running.
- You must have access to a remote system that is functional and running the IRIX operating system if you want to run the vicious socket test between two systems.
- You must specify values for all parameters in the Test Options list that are set to `NEEDED` by default.

3.11.2 Running the Vicious Socket Tests

You can run the vicious socket tests from the field_diag program.

3.11.2.1 Running the VST_PING Test

To run the VST_PING test from the field_diag program, perform the following procedure:

1. Select the VST_PING option from the Diagnostic list.
2. If you want to modify any of the parameters, perform the following actions for each parameter; otherwise, skip to Step 4.
   - Press the right arrow key to move into the Diagnostic Options list.
   - Select the parameter that you want to modify. (Table 3-5 describes the parameters that you can modify.)
   - Press the **Enter** key.
   - At the prompt `Enter New Value for the Option` in the diagnostic status area, enter the new value.
   - Press the **Enter** key.

Table 3-5 VST_PING Test User-Adjustable Parameters

Parameter	Description
`num. packets`	Specifies the number of data packets that the test should send to the remote system.
`remote host`	Specifies the host to which the test should send the data packets. You must enter this parameter.
`Pattern`	Specifies the data pattern that the test should use. Enter a hexadecimal number or any of the following settings: `bits` specifies that each 64-bit data word will have a random sequence of consecutive 1 bits. `slide0` specifies that the first 64-bit data word will have all bits (except bit 0) set to 1; bit 0 will be set to 0. Subsequent data words will be circularly left-shifted by one bit position. `slide1` specifies that the first 64-bit data word will have all bits (except bit 0) set to 0; bit 0 will be set to 1. Subsequent data words will be circularly left-shifted by one bit position. `random` specifies that data words will be set to random values. `all` specifies that the test should use built-in data patterns, one per pass, in a circular pattern.

3. Press the left arrow key to return to the Diagnostic list.
4. Press the **Enter** key to start running the VST_PING test.

   The message `Executing : VST_PING` appears in the diagnostic status area, indicating that the test is running.
5. View the diagnostic status area and Test Log Output window for messages from the VST_PING test.

3.11.2.2 Running the VST_RW Test

To run the VST_RW test from the field_diag program, perform the following procedure:

1. Select the VST_RW option from the Diagnostic list.
2. If you want to modify any of the parameters, perform the following actions for each parameter; otherwise, skip to Step 4.
   - Press the right arrow key to move into the Diagnostic Options list.
   - Select the parameter that you want to modify. (Table 3-6 describes the parameters that you can modify.)
   - Press the **Enter** key.
   - At the prompt `Enter New Value for the Option` in the diagnostic status area, enter the new value.
   - Press the **Enter** key.

Table 3-6 VST_RW Test User-Adjustable Parameters

Parameter	Description
`num. packets`	Specifies the number of data packets that the test should send to the remote system.
`remote host`	Specifies the host to which the test should send the data packets. You must enter this parameter.
`Pattern`	Specifies the data pattern that the test should use. Enter a hexadecimal number or any of the following settings: `bits` specifies that each 64-bit data word will have a random sequence of consecutive 1 bits. `slide0` specifies that the first 64-bit data word will have all bits (except bit 0) set to 1; bit 0 will be set to 0. Subsequent data words will be circularly left-shifted by one bit position. `slide1` specifies that the first 64-bit data word will have all bits (except bit 0) set to 0; bit 0 will be set to 1. Subsequent data words will be circularly left-shifted by one bit position. `random` specifies that data words will be set to random values. `all` specifies that the test should use built-in data patterns, one per pass, in a circular pattern.
`Block size`	Specifies the block size to use for I/O operations. Specify a number of bytes or `pass`. `pass` indicates that the test should set the blocksize to the current pass count value.

3. Press the left arrow key to return to the Diagnostic list.
4. Press the **Enter** key to start running the VST_RW test.

   The message `Executing : VST_RW` appears in the diagnostic status area, indicating that the test is running.
5. View the diagnostic status area and Test Log Output window for messages from the VST_RW test.

3.11.3 Output From the Vicious Socket Tests

The tests return output to the diagnostic status area of the user interface and the Test Log Output window.

3.11.3.1 Pass Output

If the VST_PING test completes without detecting any hardware failures, the diagnostic status area of the user interface displays the message `VST_PING Test Passed`.

If the VST_RW test completes without detecting any hardware failures, the diagnostic status area of the user interface displays the message `VST_RW Test Passed`.

3.11.3.2 Failure Output

If the VST_PING test detects a hardware failure or encounters an error that makes it unable to continue testing, the diagnostic status area of the user interface displays the message `VST_PING Test Failed`.

If the VST_RW test detects a hardware failure or encounters an error that makes it unable to continue testing, the diagnostic status area of the user interface displays the message `VST_RW Test Failed`.

If one of these messages appears, view the Test Log Output window for more information about the cause of the failure. Possible messages include:

- `***** Data Compare Error *****`
  `Differences written to "`*file*`".`
  `Current pass = 1`
  `Current data pattern = slide0`

  The test failed because the actual data did not equal the expected data. The failing FRU may be the board that contains the network interface that is being tested. Refer to the generated error file for more information.

  The following output shows a partial sample of the contents of an error file:

```
Executing Read Test

A : expected
B : actual
x : logical difference


A ( 0) + fe fd fb f7 ef df bf 7f fe fd fb f7 ef df bf 7f *................*
B ( 0) + 7f 06 3c 38 7c fc 1f 06 06 0c 7f ff ff ff 1c 3e *..<8|..........>*
x ( 0) + 81 fb c7 cf 93 23 a0 79 f8 f1 84 08 10 20 a3 41 *.....#.y.......A*
A ( 10) + fe fd fb f7 ef df bf 7f fe fd fb f7 ef df bf 7f *................*
B ( 10) + 70 3c 0e c0 1f 3f ff fc 07 18 f8 70 fe c0 ff ff *p<...?.....p....*
x ( 10) + 8e c1 f5 37 f0 e0 40 83 f9 e5 03 87 11 1f 40 80 *...7..@.......@.*
A ( 20) + fe fd fb f7 ef df bf 7f fe fd fb f7 ef df bf 7f *................*
B ( 20) + fe 20 7e 7f 07 7f 3e 3f ff 30 08 ff 20 20 03 78 *..~...>?.0.....x*
x ( 20) + 00 dd 85 88 e8 a0 81 40 01 cd f3 08 cf ff bc 07 *.......@........*
```

```
A ( 30) + fe fd *.............*
B ( 30) + 38 02 *8............*
x ( 30) + c6 ff *.............*
```

- ```
  Executing Ping Test
  Ping timed out
  Expected to write num bytes, sent 0 instead
  ***** Error *****
  Current pass = 1
  Current data pattern = pattern
  ```

  The VST_PING test failed because the remote system did not respond. The failing FRU may be the board that contains the network interface used to connect to the remote system.

- ```
  Executing Write Test
  connect() failed, errno = 152, text = "Connection refused".
  ```

  A write operation failed because there is no remote system to receive the data that it writes. This failure could occur because the remote system is not running or because the network interface is failing. The failing FRU may be the HIPPI-S board.

3.12 Summary of Chapter 3

This chapter described the diagnostic tests that you can run from the field_diag user interface. Table 3-7 summarizes the pass and fail messages and failing FRU for each test.

Table 3-7 Summary of Diagnostic Test Information

Diagnostic Test	Pass Message	Fail Message	Failing FRU
ATM LINC DMA verification test	`ATM Test Passed`	`ATM Test Failed`	Node board or ATM board
Network thrasher test	`network Test Passed`	`network Test Failed`	ATM, BaseIO, HIPPI-S, or MENET board
Floating-point unit (single-precision) test	`Floating point single precision Test Passed`	`Floating point single precision Test Failed`	Node board
Floating-point unit (double-precision) test	`Floating point double precision Test Passed`	`Floating point double precision test Failed`	Node board
HIPPI LINC DMA verification test	`HIPPI Test Passed`	`HIPPI Test Failed`	Node board or HIPPI-S board
MediaIO board test	`MIO Test Passed`	`MIO Test Failed`	MediaIO board
Memory test	`memory Test Passed`	`memory Test Failed`	DIMM
RAID verification test	`RAID Test Passed`	`RAID Test Failed`	Disk drive or RAID controller
SCSI thrasher test	`randomthrash Test Passed`	`randomthrash Test Failed`	BaseIO board, MSCSI board, or disk drive
Vicious HIPPI tests	`VHT_LOOPBACK Test Passed` or `VHT_REMOTE Test Passed`	`VHT_LOOPBACK Test Failed` or `VHT_REMOTE Test Failed`	HIPPI-S board
Vicious socket tests	`VST_PING Test Passed` or `VST_RW Test Passed`	`VST_PING Test Failed` or `VST_RW Test Failed`	BaseIO board, MENET board, ATM board, or HIPPI-S board