



Scan Tools
(SGI[®] Origin[®] 300 System and
SGI[®] 3000 Family)

SGI Confidential & Proprietary Information - For Internal Recipients Only

CONTRIBUTORS

Revised by Darrin Goss

Edited by Allison Gosbin

Production by Rhonda Kunsman

Engineering contributions by Karen Beighley, Kurt Kermes, and Jim Young

INFORMATION CLASSIFICATION

This document contains proprietary and confidential information of Silicon Graphics, Inc., intended for internal recipients only. The contents of this document may not be disclosed to third parties, copied, or duplicated in any form, in whole or in part, without the prior written permission of Silicon Graphics, Inc.

LIMITED RIGHTS LEGEND

The electronic (software) version of this document was developed at private expense; if acquired under an agreement with the USA government or any contractor thereto, it is acquired as "commercial computer software" subject to the provisions of its applicable license agreement, as specified in (a) 48 CFR 12.212 of the FAR; or, if acquired for Department of Defense units, (b) 48 CFR 227-7202 of the DoD FAR Supplement; or sections succeeding thereto. Contractor/manufacturer is Silicon Graphics, Inc., 1600 Amphitheatre Pkwy 2E, Mountain View, CA 94043-1351.

TRADEMARKS

Origin, SGI, the SGI logo, Silicon Graphics, and Indy are registered trademarks, and NUMAlink, Silicon Graphics Fuel, VPro, and XIO are trademarks of Silicon Graphics, Inc.

MIPS is a registered trademark of MIPS Technologies, Inc. used under license by Silicon Graphics, Inc.

Record of Revision

Version	Description
001	August 2000 Original printing.
002	May 2001 This revision corresponds to the <i>Internal Support Tools 2.4</i> CD release. It describes the new <code>brick_scan</code> tool and provides additional information about the scan hardware and the <code>scantool</code> application.
003	November 2001 This revision corresponds to the <i>Internal Support Tools 2.5.1</i> Web release. It adds information about using the scan tools to test SGI Origin 300 systems and PCI expansion modules.
004	February 2002 This revision adds information about using the scan tools to test N bricks and G bricks. It also updates the <code>brick_scan</code> description, adds FRU-to-error message mapping tables, and incorporates technical changes throughout the document.

Contents

1.	Introduction	1-1
1.1	Scan Hardware	1-3
1.1.1	SGI Origin 300 System.....	1-3
1.1.2	SGI 3000 Series System without an R brick or L2 System Controller	1-4
1.1.3	SGI 3000 Series System with an R Brick but without an L2 System Controller	1-5
1.1.4	SGI 3000 Series System with an L2 System Controller	1-6
1.1.5	SGI 3000 Series System with a G Brick	1-8
1.1.6	Scan Chains	1-9
1.2	Scan Software	1-10
1.3	Standard Test Process.....	1-10
2.	Boundary Scan Interconnect Test	2-1
2.1	About the Boundary Scan Interconnect Test	2-1
2.2	Test Algorithm	2-2
2.3	Graphical User Interface	2-4
2.4	Running the Boundary Scan Interconnect Test from brick_scan.....	2-5
2.5	Running the Boundary Scan Interconnect Test from scantest.....	2-9
2.6	Boundary Scan Interconnect Test Output	2-13
2.7	Physical and Logical Description to FRU Mapping.....	2-18
2.7.1	SGI Origin 300 Systems.....	2-18
2.7.2	SGI 3000 Series Systems	2-19
2.8	Troubleshooting Tips	2-20
2.9	Example 1: Possible PIMM Failure.....	2-21
2.10	Example 2: Possible DIMM Failure	2-23
3.	Scan-based Link-level Protocol Interconnect Test	3-1
3.1	About the Scan-based Link-level Protocol Interconnect Test.....	3-1
3.2	Test Algorithm	3-2
3.3	slit Command-line Options	3-4
3.4	Running the Scan-based Link-level Protocol Interconnect Test	3-8
3.5	Scan-based Link-level Protocol Interconnect Test Output.....	3-9

3.6	Troubleshooting Tips	3-10
3.7	Example	3-10
4.	Scripts That Run scantest and slit.....	4-1
4.1	scan_speedo2 Script.....	4-1
4.2	scan_pebrick Script.....	4-5
4.3	scan_nbrick Script.....	4-9
4.3.1	Example 1: Testing Four Null Boards in an N Brick.....	4-10
4.3.2	Example 2: Testing One Null Board in an N Brick	4-12
4.4	scan_kcar_l1 Script.....	4-14
5.	scantool Application	5-1
5.1	Available Scripts	5-1
5.2	scantool Command-line Options.....	5-3
5.3	Running Scripts from a Graphical User Interface	5-4
5.4	Running Scripts from the Command Line	5-14
5.5	scantool Script Commands	5-16
5.6	Dumping the Internal Bedrock Latches.....	5-17
5.6.1	SGI 3000 Series System C Brick.....	5-17
5.6.2	SGI Origin 300 Base Module	5-26
5.6.3	PCI Expansion Module	5-35
5.7	Dumping the Internal Router Latches	5-42

Figures

Figure 1-1	Boundary Scan Test Logic	1-1
Figure 1-2	Scan Hardware Components (SGI Origin 300 System).....	1-3
Figure 1-3	Scan Hardware Components (SGI 3000 Series System without an R Brick or L2 System Controller).....	1-4
Figure 1-4	Scan Hardware Components (SGI 3000 Series System without an L2 System Controller)	1-5
Figure 1-5	Scan Hardware Components (SGI 3000 Series System with an L2 System Controller)	1-7
Figure 1-6	Scan Hardware Components (SGI 3000 Series System with a G Brick).....	1-8
Figure 2-1	Boundary Scan Interconnect Test Algorithm Components.....	2-3
Figure 2-2	Boundary Scan Test Graphical User Interface.....	2-4
Figure 2-3	Boundary Scan Interconnect Test Warning Message	2-5
Figure 2-4	Setting the Brick Selection Parameters.....	2-6
Figure 2-5	Setting the Test Selection Parameters	2-7
Figure 2-6	Setting the Scan Communications Method Parameters	2-7
Figure 2-7	Run Button.....	2-8
Figure 2-8	Boundary Scan Interconnect Test Output (in brick_scan Interface).....	2-13
Figure 2-9	Data Pattern Indicator	2-15
Figure 2-10	Logical Description Format.....	2-16
Figure 2-11	Physical Description Format	2-17
Figure 3-1	slit Test Algorithm Components	3-3
Figure 5-1	scantool Graphical User Interface (Initial Window)	5-5
Figure 5-2	Selecting the Brick to Test.....	5-6
Figure 5-3	Selecting the Brick (Example Settings).....	5-7
Figure 5-4	Interface with Brick Selected	5-8
Figure 5-5	Selecting the Script to Run.....	5-9
Figure 5-6	Selecting a Tcl Script.....	5-10
Figure 5-7	Selecting a Tk Script.....	5-11
Figure 5-8	Running a Tcl Script	5-12
Figure 5-9	Running a Tk Script (Example Window)	5-13
Figure 5-10	Running a Tk Script (Example Output).....	5-13
Figure 5-11	scantool Graphical User Interface (Initial Window)	5-18
Figure 5-12	Configure One Brick Window	5-19

Figure 5-13	Selecting the IP35 C Brick to Use.....	5-20
Figure 5-14	Selecting the IP35 C Brick Script to Run.....	5-21
Figure 5-15	Selecting the ip35_dump_br_latches.tk Script.....	5-22
Figure 5-16	IP35 Bedrock Internal Latch Dump Window	5-23
Figure 5-17	Bedrock System Clock Warning Message	5-24
Figure 5-18	Output from the ip35_dump_latches.tk Script.....	5-25
Figure 5-19	scantool Graphical User Interface (Initial Window)	5-27
Figure 5-20	Configure One Brick Window	5-28
Figure 5-21	Selecting the SGI Origin 300 Base Module to Use.....	5-29
Figure 5-22	Selecting the SGI Origin 300 Base Module Script to Run.....	5-30
Figure 5-23	Selecting the ip45_dump_br_latches.tk Script.....	5-31
Figure 5-24	IP45 Bedrock Internal Latch Dump Window	5-32
Figure 5-25	Bedrock System Clock Warning Message	5-33
Figure 5-26	Output from the ip45_dump_latches.tk Script.....	5-34
Figure 5-27	scantool Graphical User Interface (Initial Window)	5-36
Figure 5-28	Configure One Brick Window	5-37
Figure 5-29	Choosing the PCI Expansion Module to Use.....	5-38
Figure 5-30	Selecting the PCI Expansion Module Script to Run.....	5-39
Figure 5-31	PE Brick Bedrock Internal Latch Dump Window	5-40
Figure 5-32	Output from the pe_dump_br_latches.tk Script	5-41
Figure 5-33	scantool Graphical User Interface (Initial Window)	5-43
Figure 5-34	Configure One Brick Window	5-44
Figure 5-35	Choosing the R Brick to Use.....	5-45
Figure 5-36	Selecting the R Brick Script to Run.....	5-46
Figure 5-37	Output from the r_dump_latches.tcl Script	5-47

Tables

Table 1-1	Boundary Scan Components.....	1-2
Table 1-2	SGI Origin 300 System Scan Chain Components.....	1-9
Table 1-3	SGI 3000 Series Scan Chain Components.....	1-9
Table 2-1	scantest Command-line Options	2-9
Table 2-2	FRUs Corresponding Board Type (SGI Origin 300 Systems)	2-18
Table 2-3	FRUs Corresponding Board Type (SGI 3000 Series Systems)	2-19
Table 3-1	slit Command-line Options	3-4
Table 4-1	scan_speedo2 Command-line Options	4-1
Table 4-2	scan_speedo2 Test Options	4-2
Table 4-3	scan_pebrick Command-line Options	4-5
Table 4-4	scan_pebrick Test Options.....	4-6
Table 5-1	IP35 C-brick Scripts (Located in /stand/sysco/data/scan/diags/ip35).....	5-1
Table 5-2	R-brick Scripts (Located in /stand/sysco/data/scan/diags/rbrick).....	5-2
Table 5-3	SGI Origin 300 Base Module Scripts (Located in /stand/sysco/data/scan/diags/ip45).....	5-2
Table 5-4	PCI Expansion Module Scripts (Located in /stand/sysco/data/scan/diags/pebrick).....	5-2
Table 5-5	scantool Command-line Options.....	5-3
Table 5-6	scantool Environment Variables.....	5-14
Table 5-7	scantool Configuration Files.....	5-15
Table 5-8	scantool Script Commands.....	5-16

Chapter 1

Introduction

The Institute of Electrical and Electronics Engineers (IEEE) standard 1149.1, *Standard Test Access Port and Boundary-Scan Architecture*, defines a set of test logic and related features that enable connections within and between ASICs to be tested. These features are collectively called boundary scan, scan, or JTAG.

The test logic includes a test access port (TAP), a TAP controller, an instruction register, and a group of test data registers. (Refer to Figure 1-1.)

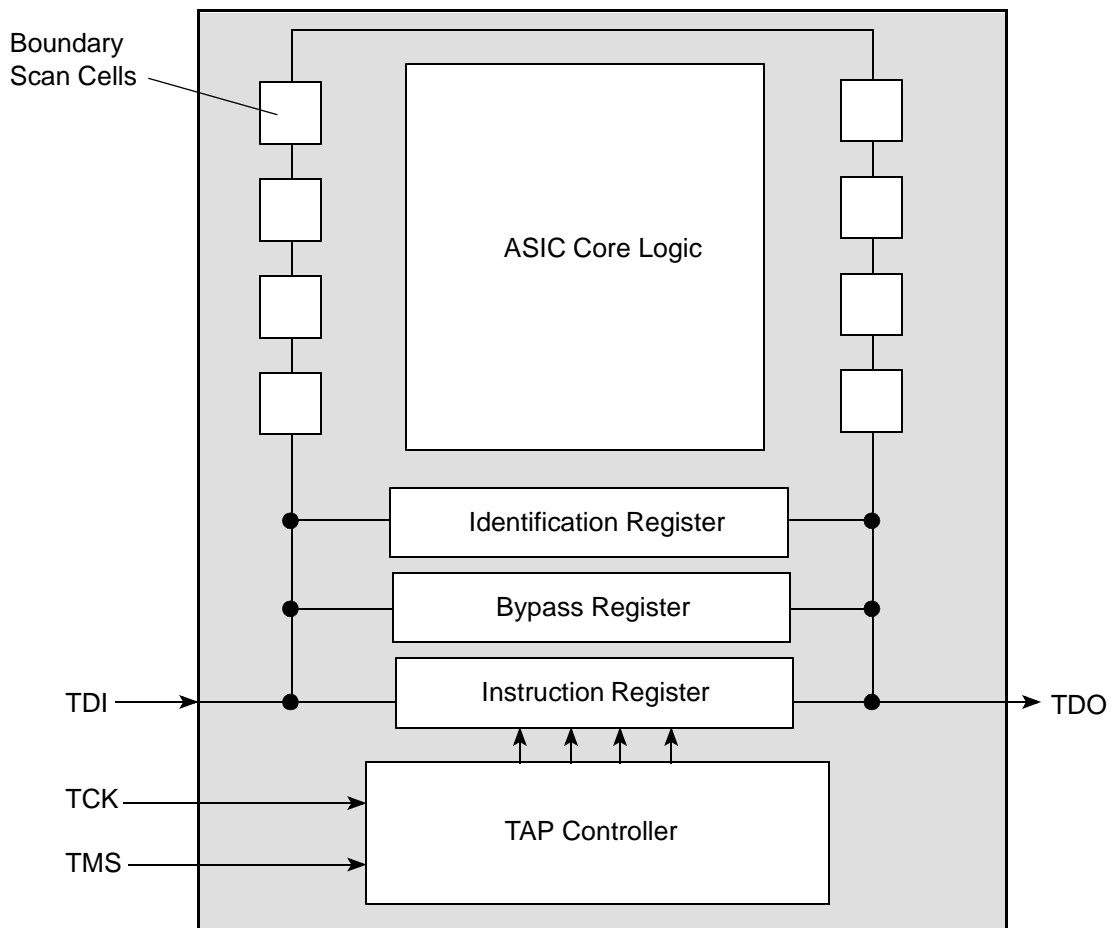


Figure 1-1 Boundary Scan Test Logic

Table 1-1 describes the boundary scan components.

Table 1-1 Boundary Scan Components

Component	Description
Test access port (TAP)	Provides access to the test logic and consist of four signal pins: test data input (TDI), test data output (TDO), test clock (TCK), and test mode select (TMS)
TAP controller	Controls the instruction register, bypass register, identification register, and boundary scan register
TMS signal	Controls test operations
TCK signal	Inputs a signal that clocks the TAP controller
TDI signal	Inputs data serially
TDO signal	Outputs data serially
Instruction register	Receives, holds, and decodes boundary scan instructions
Bypass register	Connects the TDI and TDO signals during a BYPASS instruction, which causes the data to go through the bypass register instead of the boundary scan cells (This register is used to verify integrity of the scan chain and to perform board-level testing.)
Identification (ID) register	Holds 32 bits of information that specify the manufacturer of the chip (or ASIC), the part number of the chip, and the revision of the chip ID registers are optional components; not all chips contain ID registers
Boundary scan cells	Components of the boundary scan register that are connected in a shift-register path around the boundary of the chip; these components are located between the I/O pins and the core logic, which enables the boundary scan operation to control and monitor the I/O pins
ASIC core logic	Contains the functional components of the ASIC

The following components contain ASICs that include the scan/JTAG test logic as defined by the IEEE 1149.1 standard:

- C bricks
- I bricks
- P bricks
- R bricks
- X bricks
- G bricks
- N bricks
- SGI Origin 300 base modules
- SGI Origin 300 PCI expansion modules

The diagnostic suite includes scan tests that use the scan/JTAG hardware to test connections within boards and between boards.

1.1 Scan Hardware

The following components contain hardware that is used to perform scan testing: the L1, L2, and L3 system controllers, the SGI Origin 300 base and PCI expansion modules, and C, R, I, P, X, G, and N bricks.

1.1.1 SGI Origin 300 System

In an SGI Origin 300 system (refer to Figure 1-2):

- The L3 system uses a serial or USB connection to communicate with the L1 system controller.
- The L1 system controller acts as a JTAG bus master and passes JTAG commands/data to the scan interface chip (SIC).
- The L1 system controller in the SGI Origin 300 system uses an RS-422 connection to communicate with the L1 system controller in a PCI expansion module if one is connected.
- The SIC passes the JTAG commands/data to the individual scan chains (up to four).
- The scan chains pass the commands/data to the individual ASICs. (Table 1-2 on page 1-9 shows the components on each scan chain.)

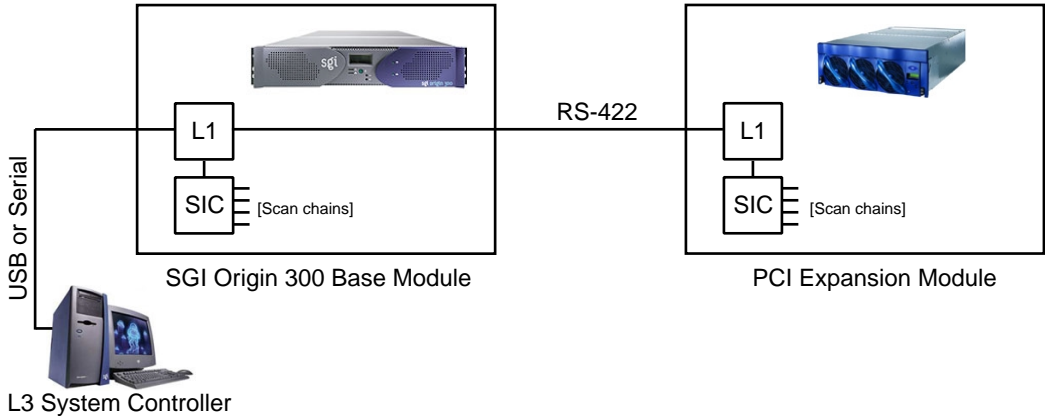


Figure 1-2 Scan Hardware Components (SGI Origin 300 System)

1.1.2 SGI 3000 Series System without an R brick or L2 System Controller

In a system without an R brick or L2 system controller (refer to Figure 1-3):

- The L3 system uses a serial or USB connection to communicate with the L1 system controller in a C brick.
- The L1 system controller in the C brick uses an RS-422 connection (through a portion of an Xtown2 or NUMAlink 3 cable) to communicate with the L1 system controllers in other I, P, X, N, and C bricks.
- The L1 system controller in each C, R, I, P, X, and N brick acts as a JTAG bus master and passes JTAG commands/data to the scan interface chip (SIC) in each brick.
- The SIC in each brick passes the JTAG commands/data to the individual scan chains (up to four) in the brick.
- The scan chains pass the commands/data to the individual ASICs in the bricks. (Table 1-3 on page 1-9 shows the components on each scan chain.)

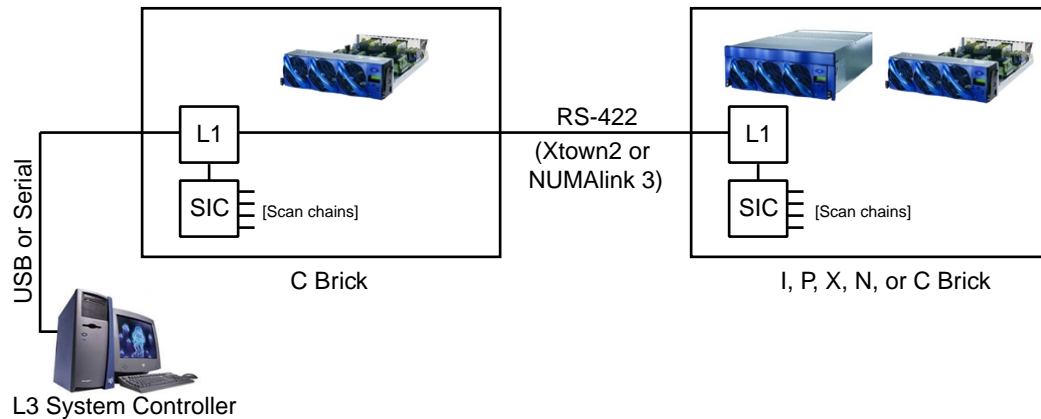


Figure 1-3 Scan Hardware Components (SGI 3000 Series System without an R Brick or L2 System Controller)

1.1.3 SGI 3000 Series System with an R Brick but without an L2 System Controller

In a system that has an R brick but does not have an L2 system controller (refer to Figure 1-4.):

- The L3 system controller uses a serial or USB connection to communicate with the L1 system controller in a C brick or a USB connection to communicate with the L1 system controller in an R brick.
 - If the L3 system controller is connected to a C brick, the L1 system controller in the C brick uses an RS-422 connection (through a portion of an Xtown2 or NUMAlink 3 cable) to communicate with the L1 system controllers in other I, P, X, N, and C bricks.
 - If the L3 system controller is connected to an R brick, the L1 system controller in the R brick uses an RS-422 connection (through a portion of a NUMAlink 3 cable) to communicate with the L1 system controller in another R brick or a C brick.
- The L1 system controller in each C, R, I, P, X, and N brick acts as a JTAG bus master and passes JTAG commands/data to the SIC in each brick.
- The SIC in each brick passes the JTAG commands/data to the individual scan chains (up to four) in the brick.
- The scan chains pass the commands/data to the individual ASICs in the bricks. (Table 1-3 on page 1-9 shows the components on each scan chain.)

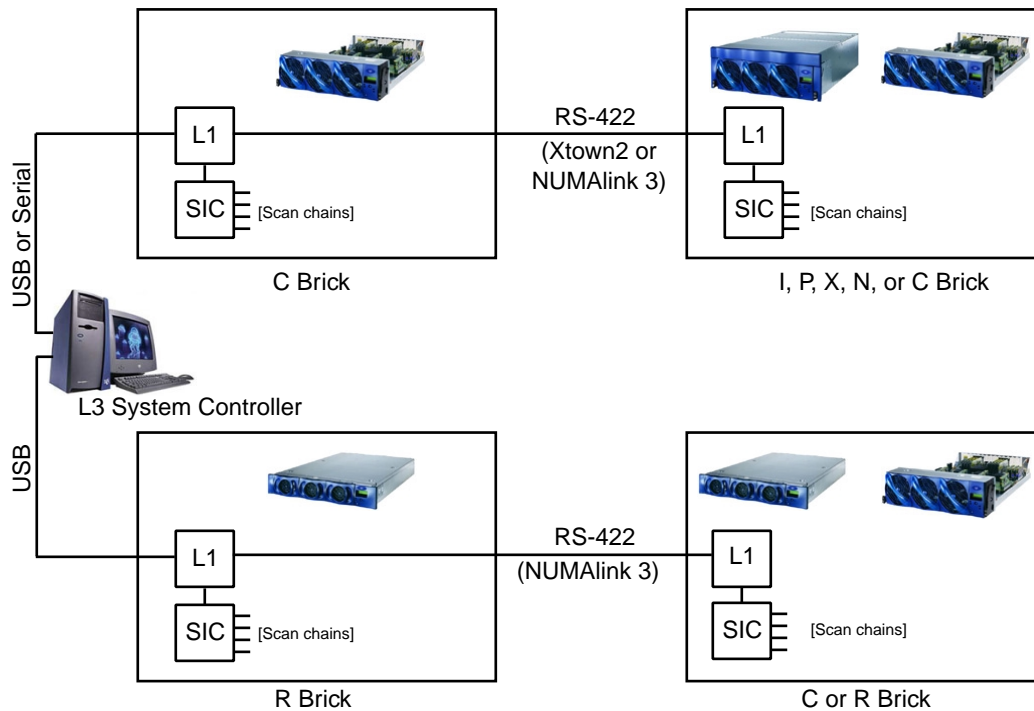


Figure 1-4 Scan Hardware Components (SGI 3000 Series System without an L2 System Controller)

1.1.4 SGI 3000 Series System with an L2 System Controller

In a system with an L2 system controller (refer to Figure 1-5):

- The L3 system controller uses an Ethernet connection to communicate with the L2 system controller.
 - Note:** If necessary, the L3 system controller can bypass the L2 system controller and use a serial or USB connection to communicate directly with the C and R bricks (as in a system without an L2 system controller).
- The L2 system controller uses USB connections to communicate with the L1 system controllers in the C bricks and R bricks.
 - If the L2 system controller is connected to a C brick, the L1 system controller in the C brick uses an RS-422 connection (through a portion of an Xtown2 or NUMALink 3 cable) to communicate with the L1 system controllers in other I, P, X, N, and C bricks.
 - If the L2 system controller is connected to an R brick, the L1 system controller in the R brick uses an RS-422 connection (through a portion of a NUMALink 3 cable) to communicate with the L1 system controller in another R brick or a C brick.
- The L1 system controller in each C, R, I, P, X, and N brick acts as a JTAG bus master and passes JTAG commands/data to the SIC in each brick.
- The SIC in each brick passes the JTAG commands/data to the individual scan chains (up to four) in the brick.
- The scan chains pass the commands/data to the individual ASICs in the bricks. (Table 1-3 on page 1-9 shows the components on each scan chain.)

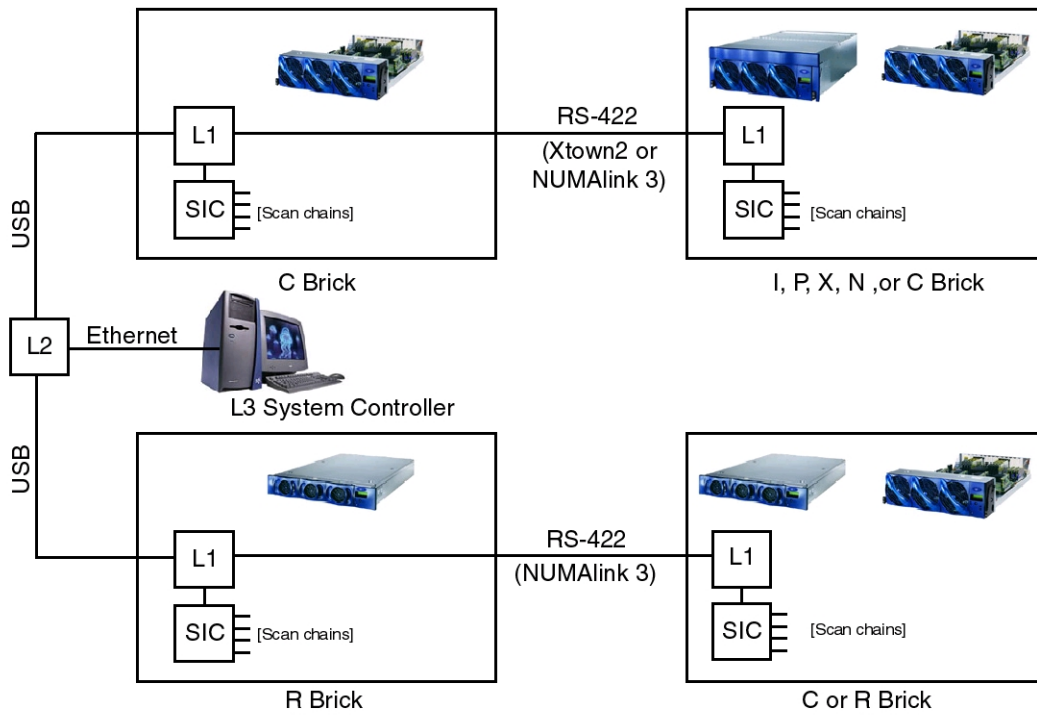


Figure 1-5 Scan Hardware Components (SGI 3000 Series System with an L2 System Controller)

1.1.5 SGI 3000 Series System with a G Brick

In a system with a G brick (refer to Figure 1-6):

- The L3 system controller uses an Ethernet connection to communicate with the L2 system controller.
- The L2 system controller uses USB connections to communicate with the L1 system controllers in the G brick.
- The L1 system controller passes JTAG commands/data to the SIC in the brick.
- The SIC passes the JTAG commands/data to the individual scan chains (up to four) in the brick.
- The scan chains pass the commands/data to the individual ASICs in the bricks.
(Table 1-3 on page 1-9 shows the components on each scan chain.)

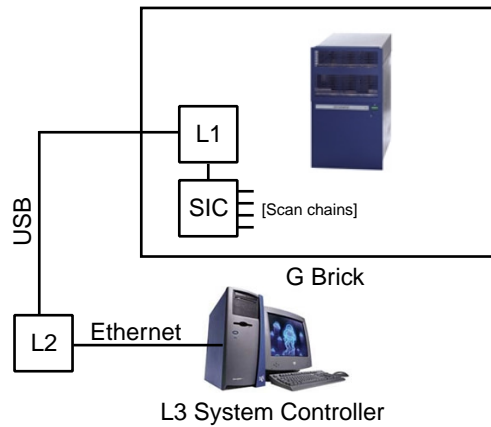


Figure 1-6 Scan Hardware Components (SGI 3000 Series System with a G Brick)

1.1.6 Scan Chains

Table 1-2 lists the components in each scan chain for SGI Origin 300 systems.

Table 1-2 SGI Origin 300 System Scan Chain Components

SIC	Chain 0	Chain 1	Chain 2	Chain 3
0	IO8	PCI1	PCI2	Not used
1	CPUA CPUB CPUA1 CPUB1	SRAMs: SCA_T SCA_1 ... SCB1_3 SCB1_2	Bedrock Xbridge	DIMM0 DIMM1 DIMM2 DIMM3 DIMM4

Table 1-3 lists the components in each scan chain for the various bricks in SGI 3000 series systems.

Table 1-3 SGI 3000 Series Scan Chain Components

Brick	Chain 0	Chain 1	Chain 2	Chain 3
C (IP35)	PIMM0	PIMM1	Bedrock	DIMM0 DIMM1 DIMM2 DIMM3 DIMM4 DIMM5 DIMM6 DIMM7
I	Xbridge	IOC3	PCI11 PCI12 PCI13 PCI21 PCI22	Not used
P	Not used	PCI41 PCI42 PCI51 PCI52 PCI61 PCI62	PCI11 PCI12 PCI21 PCI22 PCI31 PCI32	XBG2 XBG0 XBG1
R	Router	Not used	Not used	Not used
X	Xbridge	PORT8 (XIO slot) PORT9 (XIO slot)	PORTC (XIO slot) PORTD (XIO slot)	Not used
G	GE (pipe 1) (kcar_ge)	Ktown2	GE (pipe 0) (kcar_ge)	Not used
N	Null board 0	Null board 1	Null board 2	Null board 3

1.2 Scan Software

The diagnostic suite includes two scan tests:

- Boundary scan interconnect test

The boundary scan interconnect test manipulates the boundary scan registers in various chips throughout the system to verify system interconnections. It applies a set of test vectors to the system to detect physical connection defects (for example, stuck-at faults and shorts).

- Scan-based link-level protocol interconnect test

The scan-based link-level protocol interconnect test uses additional scan-accessible registers to exercise brick-to-brick connections at-speed. For SGI Origin 300 systems, it tests on-board links and connections (regular and loopback) between base modules and PCI expansion modules.

For SGI 3000 series systems, it tests the connectors on C, R, I, P, X, G, and N bricks and the cables between these bricks.

Note: The scan-based link-level protocol test runs only in components that use LLP (router, Bedrock, and Xbridge).

The diagnostic suite also includes a scripting tool, `scantool`. `scantool` enables you to control scan operations from Tcl/Tk scripts.

The scan software resides on the L3 system controller. To use the scan software, you must have an L3 system controller, a laptop loaded with the L3 software, or a remote support connection to a system.

1.3 Standard Test Process

Use the following process to test a system with the scan tools:

1. Install the cables (normal or loopback) for the configuration that you want to test.
2. Log into the L3 system controller as `sgidiag`.
3. Start the L2 system controller (if necessary), and power up the hardware.
4. Run the boundary scan interconnect test to verify the scan hardware and verify connections within bricks.
5. Run the scan-based link-level protocol interconnect test to verify connections between bricks.

Chapter 2

Boundary Scan Interconnect Test

This chapter describes the boundary scan interconnect test.

2.1 About the Boundary Scan Interconnect Test

The boundary scan interconnect test verifies connections within SGI Origin 300 systems and within C, R, I, X, P, G, and N bricks in SGI 3000 series systems. Use it to verify proper operation of any components that you add to a system.

The boundary scan interconnect test uses the following applications:

- A test application (`scantest`) that manipulates the boundary scan registers in various chips of a brick to verify connections within the brick. `scantest` applies a set of test vectors to a brick to detect physical connection defects (stuck-at faults and shorts).
- A graphical user interface application (`brick_scan`) that enables you to set test parameters and view test output.

Note: Several scripts are available to run the boundary scan interconnect test and scan-based link-level protocol interconnect test. Refer to Chapter 4, “Scripts That Run `scantest` and `slit`,” for more information.

When the boundary scan interconnect test detects an error, it identifies the chips and pins on the failing net and also displays the actual and expected data.

2.2 Test Algorithm

The boundary scan interconnect test uses the following test algorithm:

1. Load the chain configuration and reference data.
2. Load the reference data for each board definition.
3. Initialize JTAG communications to the selected scan controller(s).
4. Initialize the chain TAPs and configure the SIC(s).
5. Perform comprehensive scan hardware integrity tests to verify that the scan hardware is functional enough to perform boundary scan interconnect testing:
 - Perform the instruction register length test.
 - Perform the instruction register path test.
 - Perform the bypass register length test.
 - Perform the bypass register path test.
 - Perform the chain-level boundary register length test.
 - Perform the chain-level boundary register path test.
6. Perform the board-level boundary scan register interconnect test.
7. Print a program termination status message.

Caution: Each step must pass before the next step can be run. If the integrity tests in Step 5 do not pass, the interconnect test does not run because the results would be invalid. Do not manually run the interconnect test until the integrity tests pass.

Figure 2-1 shows the output from each step of this algorithm.

```

Boundary-Scan Test Software (scantest V3.11) Mon Apr  2 11:37:41 2001

1 Loading chain configuration and reference data
    File: /stand/sysco/cfg/scan/c001dimmm04.cfg

2 Loading reference data for each board definition
    Path: /stand/sysco/data/scan/cmp

3 Initializing JTAG communications to scan controller(s)
    No hardware; running in software simulation/emulation mode

4 Initializing chain TAPs and configuring SICs:
    001c01 with SIC at address 0x00 (CER=0x0C)

Testing Integrity of Boundary Scan Instruction Registers
Performing IR length test on active UUTs:
001c01 SIC 0x0 ... passed (chain length=149)

Performing IR path test on active UUTs:
001c01 SIC 0x0 ... passed

Testing Integrity of Boundary Scan Bypass Registers
Performing BYPASS length test on selected UUTs:
5 001c01 SIC 0x0 ... passed (chain length=19)

Performing BYPASS path test on selected UUTs:
001c01 SIC 0x0 ... passed

Testing Integrity of Boundary Scan Data Registers
Performing chain-level BOUNDARY length test on selected UUTs:
001c01 SIC 0x0 ... passed (chain length=1706)

Performing chain-level BOUNDARY path test on selected UUTs:
001c01 SIC 0x0 ... passed

6 Performing system-level interconnect test on given configuration:
    (22 of 22 patterns will be applied)
    ++++++
    Errors were detected during interconnect test.

    Failure information for BSR interconnect test:

    Module and raw position : 000-000022
    Expected data           : 1010101010010101010101
    Actual data             : 0000000000000000000000
    Difference data         : ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^
    Driving node index      : 00000000000000000001100
    Logical description     : [001c01:NODE:BR:MB_DATA_43_BUS]
    001c01:DIMM0:U2:A4
    Physical description    : [001c01:NODE:J0J2:AD35] 001c01:DIMM0:A9B3:1

    Number of failing signals displayed from test: 1

7 Normal Program Termination

```

Figure 2-1 Boundary Scan Interconnect Test Algorithm Components

2.3 Graphical User Interface

The `brick_scan` graphical user interface enables you to set test parameters and view test output. (Refer to Figure 2-2.)

The `Brick Selection` parameters specify the brick location (rack and slot) and type of brick to test.

Use the following brick type settings for SGI 3000 series systems: IP35, IP37, I, R, P, and X. If you select an IP35 C brick, you can also select which PIMMs to test (PIMM0 and/or PIMM1) and the CPU type (MIPS R12000 or R14000 CPUs). The IP37 setting is not valid for field use.

Use the `Speedo2` brick type setting for SGI Origin 300 systems. You can specify the number of CPUs in the system (2 or 4) and specify whether the IO8 board should be tested.

Use the `Asterix` brick type setting for Silicon Graphics Fuel workstations. You can specify whether the VPro graphics board should be tested (with the `Odyssey` setting).

The screenshot shows the `Brick Scan 1.4` GUI. It features a `Brick Selection` panel with fields for `Rack Number` (1), `Slot Number` (21), and `Brick Type` (I). A `Test Selection` panel on the right lists several test options with checkboxes, all of which are checked. Below this is a `Loop` field set to 0 and `Select All` / `Deselect All` buttons. The `L2 Address` field is set to 10.17.160.98. At the bottom, there are `Help`, `Run`, and `Exit` buttons. A large text area at the bottom displays the test output, including messages like "Performing BYPASS path test on selected UUTs: 001i21 SIC 0xA ... passed" and "No errors detected during interconnect test." highlighted in green.

The `Test Selection` parameters specify the tests to run.

The `Loop` parameter specifies the number of times to repeat the tests. (Default = 0 = run 1 time)

The `L2 Address` parameter specifies the IP address of the L2 controller or L2 emulator for the system to test.

`brick_scan` automatically sets this parameter. (You can use the `L2 cfg` command or the `L3 l2find` command to determine the IP address.)

The command buttons display help information, run the selected tests, and exit the graphical user interface.

The test output area provides detailed information about the activities that the boundary scan interconnect test performed and error messages for any failures that it detected.

Figure 2-2 Boundary Scan Test Graphical User Interface

2.4 Running the Boundary Scan Interconnect Test from brick_scan

Warning: Ensure that the system is offline before you run the boundary scan interconnect test. If the operating system is running when you run this test, the operating system will crash and customer data will be lost.

Perform the following procedure to run the boundary scan interconnect test on a single brick from the `brick_scan` graphical user interface:

1. Power up the system.
2. Log into the L3 system controller as `sgidiag` (default password: `sgi!diag`).
3. Enter the `cd /stand/sysco/usr` command to change to the `/stand/sysco/usr` directory.
4. Connect to the L2 system controller.

- To use a hardware L2 connection:

Connect the L3 system controller to the L2 system controller with an Ethernet connection by using a crossover cable.

Obtain the IP address of the L2 system controller:

```
cat /var/state/dhcp/dhcpd.leases
```

Enter the command: `/stand/sysco/l2term --l2 <host>`

- To use the L2 emulator:

Enter the command: `/stand/sysco/bin/l2`

5. Enter the `/stand/sysco/bin/brick_scan` command to start the `brick_scan` application.

The `brick_scan` application displays a warning message. (Refer to Figure 2-3.) Review this information and ensure that the system is ready for testing; then, click Continue.

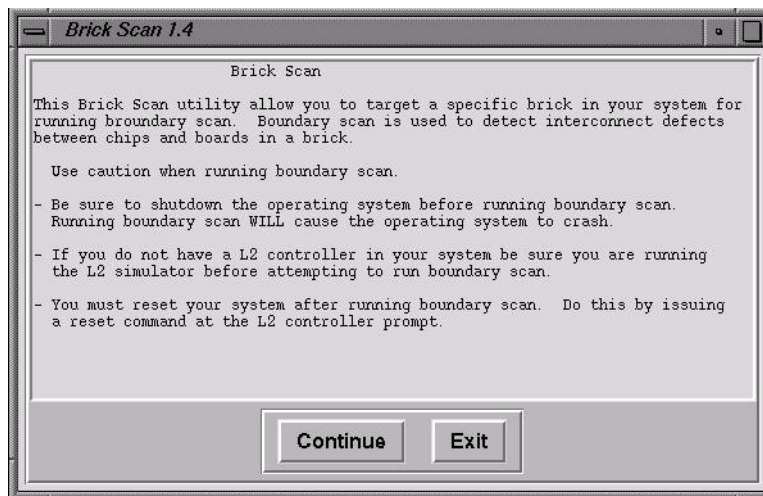


Figure 2-3 Boundary Scan Interconnect Test Warning Message

6. Select the brick that you want to test (refer to Figure 2-4):

- Enter the rack number in the `Rack Number` field.
- Enter the slot number in the `Slot Number` field.

If you do not know the rack and slot locations of the brick that you want to test, use the `L2 cfg` command to determine the bricks that are available.

- Select the brick type from the `Brick Type` menu.

Use the `IP35`, `IP37`, `I`, `R`, `P`, and `X` brick type settings to test SGI 3000 series bricks. If you are testing an `IP35 C` brick, you can select the PIMMs to test (none, `PIMM0`, and/or `PIMM1`) and specify the type of CPUs in the system (`MIPS R12000` or `MIPS R14000` CPUs).

Note: The `IP37` setting is not valid for field use.

Use the `Speedo2` brick type setting to test an SGI Origin 300 system. You can specify the number of CPUs in the system and specify whether the `IO8` board should be tested.

Use the `Asterix` brick type to test a Silicon Graphics Fuel visual workstation. You can specify whether the `VPro` graphics card should be tested by selecting or deselecting the `Odyssey` setting. Refer to the *Silicon Graphics Fuel Visual Workstation Diagnostic Reference Manual*, publication number 108-0350-00x, for more information about testing a Silicon Graphics Fuel visual workstation.

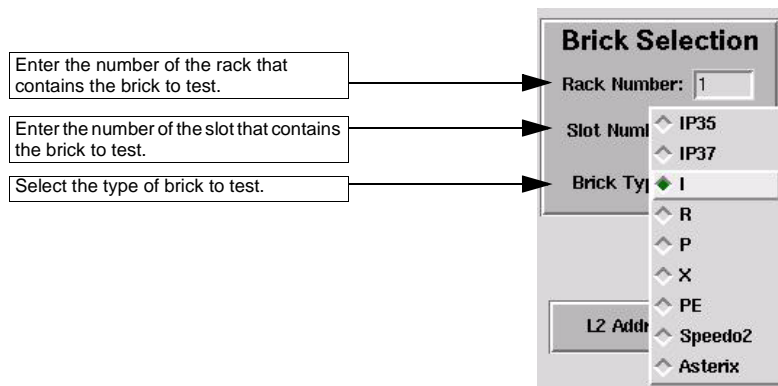


Figure 2-4 Setting the Brick Selection Parameters

7. Select the test sections that you want to run. (Refer to Figure 2-5.) SGI recommends that you run all test sections.

Caution: Each test section must pass before the next section can be run. If the integrity tests do not pass, the interconnect test does not run because the results would be invalid. Do not manually run the interconnect test until the integrity tests pass.

If you wish to run the test multiple times, change the `Loop` parameter, and click on the box next to the `Loop` parameter. The test defaults to one pass.

Note: If you set the `Loop` parameter but do not click the box next to it, the test performs only one pass. If you want to run multiple passes, you must set the `Loop` parameter and click on the box next to it.

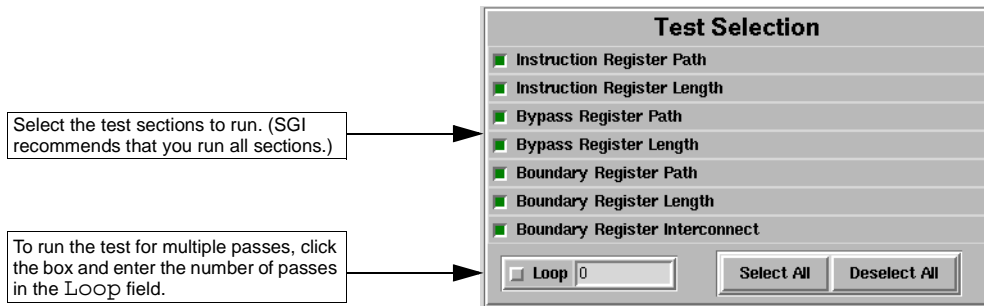


Figure 2-5 Setting the Test Selection Parameters

8. Modify the `L2 Address` parameter, if necessary. (Refer to Figure 2-6.)

The `L2 Address` parameter specifies the IP address of the L2 controller or L2 emulator for the system to test.

Note: The `brick_scan` application defaults to the first L2 system controller that it finds. If the `L2 address` field does not contain the IP address of the L2 system controller that you want to use, use the `L2 cfg` command or the `L3 l2find` command to determine the correct IP address.

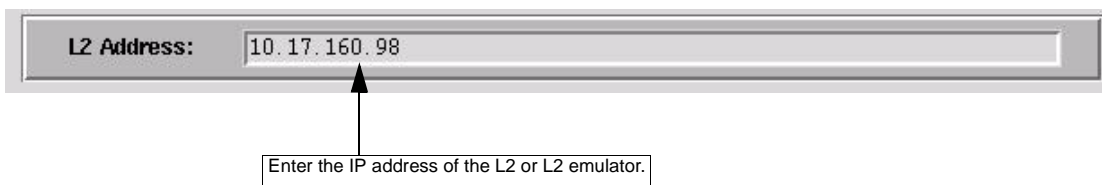


Figure 2-6 Setting the Scan Communications Method Parameters

9. Click on the `Run` button. (Refer to Figure 2-7.)

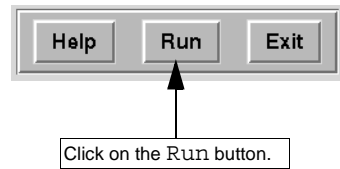


Figure 2-7 Run Button

10. Interpret the output. (Refer to Section 2.6, “Boundary Scan Interconnect Test Output.”)

Note: The boundary scan interconnect test saves the test output in the `scantest.dmp` file in the current directory; however, it overwrites this file each time you run the test. If you want to save the output so you can analyze it later, copy the contents of the file to a new file before you run the boundary scan interconnect test again.

11. When you are done testing, enter the `L2 reset` command to reset the system.

2.5 Running the Boundary Scan Interconnect Test from scantest

scantest uses the following command syntax:

```
scantest [-h] <-f file> [-ID num] [-d file] [-e num] [-p num] [-t test] [-m mode]
[-l level] [-C] [-s mode] [-v] [--l2 <host> | --interface <iface> |
--ssn <serial>[:<rack>] | --sysname <system_name>[:<rack>] | --scdev <device> |
--serial <host>:<port> | --dev <device_name>]
```

Table 2-1 describes the command-line options that you can use to control scantest.

Table 2-1 scantest Command-line Options

Option	Description
-h	Displays the available command-line options and then exits.
-f <file>	<p>Specifies a configuration file to use. The configuration file defines the hardware to test. (Default: <code>slit.cfg</code>)</p> <p>Use the following configuration files to test all bricks in a system:</p> <p><code>sys8p.cfg</code> = system with 8 processors <code>sys16p.cfg</code> = system with 16 processors <code>sys32p.cfg</code> = system with 32 processors <code>sys64p.cfg</code> = system with 64 processors <code>sys128p.cfg</code> = system with 128 processors</p> <p>For example, to test all bricks in a 16-processor system, enter the following command:</p> <pre>scantest -f /stand/sysco/cfg/scan/sys16p.cfg</pre> <p>Use the following configuration files to test single bricks in a system:</p> <p><code>c001.cfg</code> = IP35 C brick <code>i002.cfg</code> = I brick <code>p001.cfg</code> = P brick <code>r001.cfg</code> = R brick <code>x001.cfg</code> = X brick <code>n001.cfg</code> = N brick <code>g001.cfg</code> = G brick <code>ip45_001.cfg</code> = IP45 motherboard and CPUs (SGI Origin 300 system)</p> <p>For example, to test the R brick that is connected to the L2 system controller:</p> <pre>scantest -f /stand/sysco/cfg/scan/r001.cfg</pre> <p>Other configuration files are available in the <code>/stand/sysco/cfg/scan/</code> directory. Refer to the <code>/stand/sysco/cfg/scan/README.cfg</code> file for more information.</p> <p>The ID field in each configuration file describes the brick to test. By default, the ID field is set to zero (0), which selects the local brick. Use the following formula to select a different brick:</p> $ID = (\text{rack} * 64) + \text{slot}$ <p>For example, to test an I/O brick at location 002i10 (rack =2, slot = 10), set the ID field to $(2 * 64) + 10$, which equals 138.</p> <p>If you want to modify a configuration file, copy it to a directory for which you have write permission. If you modify the ID field, you must set the first number of the chain definition to the same value (for example, <code>START CHAIN 138</code>).</p>

Table 2-1 (continued) scantest Command-line Options

Option	Description
-ID <num>	<p>Overrides the ID field listed in a configuration file. This enables you select a different brick for testing without modifying the configuration file. Use the following formula to select a different brick:</p> $\text{ID} = (\text{rack} * 64) + \text{slot}$ <p>For example, to test a P brick in slot 7 of rack 2, enter the following command:</p> <pre>scantest -f /stand/sysco/cfg/scan/p001.cfg -ID 135</pre>
-d <file>	<p>Specifies the output file that receives output from the test. (Default: scantest.dmp)</p>
-e <num>	<p>Defines the maximum number of errors that the test displays. (Valid values are between 0 [display only pass/fail information] and 10,000; default: 200)</p>
-p <num>	<p>Specifies the number of test passes that scantest should perform. (Valid values are between 1 and 10,000; default: 1)</p>
-t <test>	<p>Specifies the tests to run.</p> <p>Valid tests:</p> <ul style="list-style-type: none"> irp = instruction register path test irl = instruction register length test ir = instruction register path test and instruction register length test prp = bypass register path test prl = bypass register length test pr = bypass register path test and bypass length test brp = boundary register path test brl = boundary register length test br = boundary register path test and boundary register length test int = boundary register interconnect test id = read and display device ID registers all = all tests <p>Use + to select multiple tests (for example: ir+br+int).</p> <p>The order that the tests are specified in is irrelevant. Certain tests depend on other tests completing successfully, so selecting one test may cause other tests to be automatically selected.</p> <p>(Default: ir+pr+br+int)</p>
-m <mode>	<p>Specifies the mode for register tests.</p> <p>Valid modes:</p> <ul style="list-style-type: none"> normal = normal lengths and minimal pattern(s) elen = extended lengths epat = extended pattern set <p>If normal is specified, then no other option is valid.</p> <p>(Default: normal)</p>

Table 2-1 (continued) scantest Command-line Options

Option	Description
-l <level>	Specifies the level of the register tests. Valid modes: chn = chain level register test brd = board level register test chip = chip level register test (Default: chn)
-C	Selects an individual Scan Interface Chip (SIC) to run the test. (The test will not run subordinate chains/hardware.) When you use this option: You can specify the irl, irp, ir, prl, prp, or pr tests with the -t option. The -l option is ignored. The -m option is used for lengths and patterns.
-s <mode>	Selects a stepping mode. Valid modes: no = no test stepping; run continuous all = step through each pattern; prompt user err = step through failing pattern; prompt user (Default: no)
-v	Selects verbose mode, which writes additional messages to the output.
--l2 <host>	Connects to the L2 system controller with the specified host name or IP address
--interface <iface>	Connects to the L2 system controller with the specified network interface (for example, eth0)
--ssn <serial> [: <rack>]	Connects to the system with the specified serial number (for example, L01234567)
--sysname <system_name> [: <rack>]	Connects to the system specified by <system_name>
--scdev <device>	Connects to the specified system controller
--serial <host> : <port>	Connects to the specified Ethernet host name and port
--dev <device_name>	Connects to the specified local serial port device

Running the boundary scan interconnect test with the `scantest` command enables you to manually control the settings that the test uses. It also enables you to run the boundary scan interconnect test on multiple bricks.

Warning: Ensure that the system is offline before you run the boundary scan interconnect test. If the operating system is running when you run this test, the operating system will crash and customer data will be lost.

Perform the following procedure to run the boundary scan interconnect test on one or more bricks with the `scantest` command:

1. Power up the system.
2. Log into the L3 system controller as `sgidiag` (default password: `sgi!diag`).
3. Enter the `cd /stand/sysco/usr` command to change to the `/stand/sysco/usr` directory.
4. Connect to the L2 system controller.
 - Using a hardware L2 connection:
Connect the L3 system controller to the L2 system controller with an Ethernet connection by using a crossover cable.
Obtain the IP address of the L2 system controller:

```
cat /var/state/dhcp/dhcpd.leases
```


Enter the command: `/stand/sysco/l2term --l2 <host>`
 - Using the L2 emulator:
Enter the command: `/stand/sysco/bin/l2`
5. Enter the appropriate command to start the boundary scan interconnect test:
 - For the hardware L2 connection:

```
/stand/sysco/bin/scantest -f /stand/sysco/cfg/scan/<file> --l2 <host> [test_options]
```


Example: `scantest -f /stand/sysco/cfg/scan/p001.cfg --l2 10.1.1.37`
 - For the L2 emulator:

```
/stand/sysco/bin/scantest -f /stand/sysco/cfg/scan/<file> [test_options]
```


Example: `scantest -f /stand/sysco/cfg/scan/p001.cfg`

Refer again to Table 2-1 for descriptions of the available command-line options.

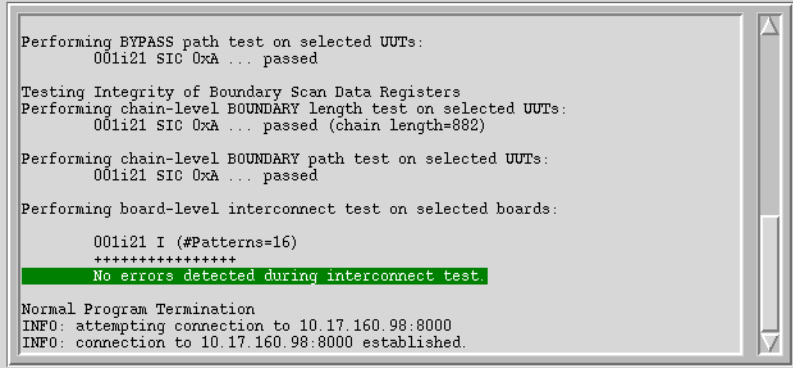
Note: The boundary scan interconnect test saves the test output in the `scantest.dmp` file in the current directory; however, it overwrites this file each time you run the test. If you want to save the output so you can analyze it later, copy the contents of the file to a new file before you run the boundary scan interconnect test again.
6. Interpret the output. (Refer to Section 2.6, “Boundary Scan Interconnect Test Output.”)
7. When you are done testing, enter the `L2 reset` command to reset the system.

2.6 Boundary Scan Interconnect Test Output

The boundary scan interconnect test returns output in the following ways:

- `scantest` returns output to the shell where you execute it.
- `brick_scan` displays test output in the lower portion of the interface. (Refer to Figure 2-8.)

The test output area provides detailed information about the activities that the test performed and error messages for any failures that it detected. Failure and error messages are highlighted in red, and pass messages are highlighted in green.



```
Performing BYPASS path test on selected UUTs:
001i21 SIC 0xA ... passed

Testing Integrity of Boundary Scan Data Registers
Performing chain-level BOUNDARY length test on selected UUTs:
001i21 SIC 0xA ... passed (chain length=882)

Performing chain-level BOUNDARY path test on selected UUTs:
001i21 SIC 0xA ... passed

Performing board-level interconnect test on selected boards:

001i21 I (#Patterns=16)
*****
No errors detected during interconnect test.

Normal Program Termination
INFO: attempting connection to 10.17.160.98:8000
INFO: connection to 10.17.160.98:8000 established.
```

Figure 2-8 Boundary Scan Interconnect Test Output (in `brick_scan` Interface)

The boundary scan interconnect test returns the following error messages:

```
ERROR - Unable to open output file for writing.
File: scantest.dmp
Writing output data to screen instead of file
```

This message appears when you do not have permission to create the `scantest.dmp` output file. The boundary scan interconnect test continues to run, but it does not send output to a file. To save test output to a file, run the test from a directory where you have write permission.

```
Unable to run interconnect test because of register test error(s).
```

This message appears when one of the boundary scan validation tests fails. When this happens, the interconnect test does not run. Review the output that precedes this message to determine the failing register:

- If the failing register is an instruction register, verify that the brick is powered up. If the brick is powered up, there is a failure in the boundary scan chain. If the brick is not powered up, power it up and rerun the test.
- If the failing register is not an instruction register, there is a failure in the boundary scan chain.

```
ERROR - scantest V3.11 (snl3sc.c @ 280, code=0)
Unable to read scan response packet from controller.
Unknown Module 0, message = -1
Confirm configuration of brick specified with ID=483 (rack=7, slot=35)
```

This message typically appears when you specify incorrect rack and slot numbers for the brick to test. Verify the rack and slot number of the brick you want to test and rerun the test.

This message also appears if the brick is completely powered off (the L1 system controller is not running). Power on the brick and rerun the test.

Errors were detected during interconnect test.

This message appears when the boundary scan interconnect test detects a failing component in the brick. Review the output following this message to determine the failing component.

After displaying this message, the test continues; when the testing is completed, the test generates a list of all chain positions that had failing data and identifies the failing chips and pins by physical and logical locations. The detailed failure information uses the following format:

```
BOUNDARY REGISTER INTERCONNECT FAILURE
-----

Chain and raw position : 000-000601
Expected data         : 0110011010010101
Actual data           : 0000000000000100
Difference data        :  ^^  ^^ ^  ^  ^  ^
Driving node index    : 111111111110011
Logical description   : [101i01:I:XBG0:PF_PCI_SERR_N]
101i01:I:IOC3:PCI_SERR_L
Physical description  : [101i01:I:H7F9:AD21] 101i01:I:J5D4:N2
```

The failure information contains the following data:

- Chain and raw position: indicate the location where the test detected a miscompare. The format of this value is <chain number>-<bit position>. (For example, 000-000601 indicates that the test detected the miscompare on chain number 0 at bit position 601.)
- Expected data: indicates the signature that the test expects to receive for the data patterns that it is using.
- Actual data: indicates the actual signature that was sensed for the data patterns.
- Difference data: contains a ^ symbol for each pattern in the signature where the actual data did not match the expected data.

Note: The number of data patterns that the test uses depends on the number of nets being tested. Two types of patterns exist: patterns to detect *stuck-at* failures and patterns to detect *short* failures. For the Expected data, Actual data, and Difference data values, each 0/1 value represents a pattern, with the first pattern (pattern 0) starting at the right and the pattern numbers incrementing by 1 as they go to the left (refer to Figure 2-9).

```

Expected data      : 0110011010010101
Actual data       : 0000000000000100
Difference data   :  ^^  ^^  ^  ^  ^
                   |  |  |
                   |  | pattern 0
                   |  | pattern 2
                   |  | pattern 4

```

Figure 2-9 Data Pattern Indicator

- Driving node index: indicates which node on the net is *driving* the transfer for each test pattern. (The driving node sends the data pattern to the receiving node.)

The node index value is a single digit that specifies which node is the driving node for the data pattern. The digit starts at 0 for the first node on the left side of the Logical description and Physical description lists and then increments by 1 for each successive node. The node index value displays an x if a net is not valid for a pattern (for example, no driver is available).

A net may have more than one driver if bidirectional signals are available on the net. If possible, *stuck-at* patterns are transferred from all potential drivers; *short* patterns are applied using a single driver.

- Logical description: provides a logical description of the nodes on the failing net. The format of each node is <chain and brick>:<board>:<chip>:<pin>. (Refer to Figure 2-10.) The boundary scan interconnect test displays the node that detected the miscompare in square brackets [].

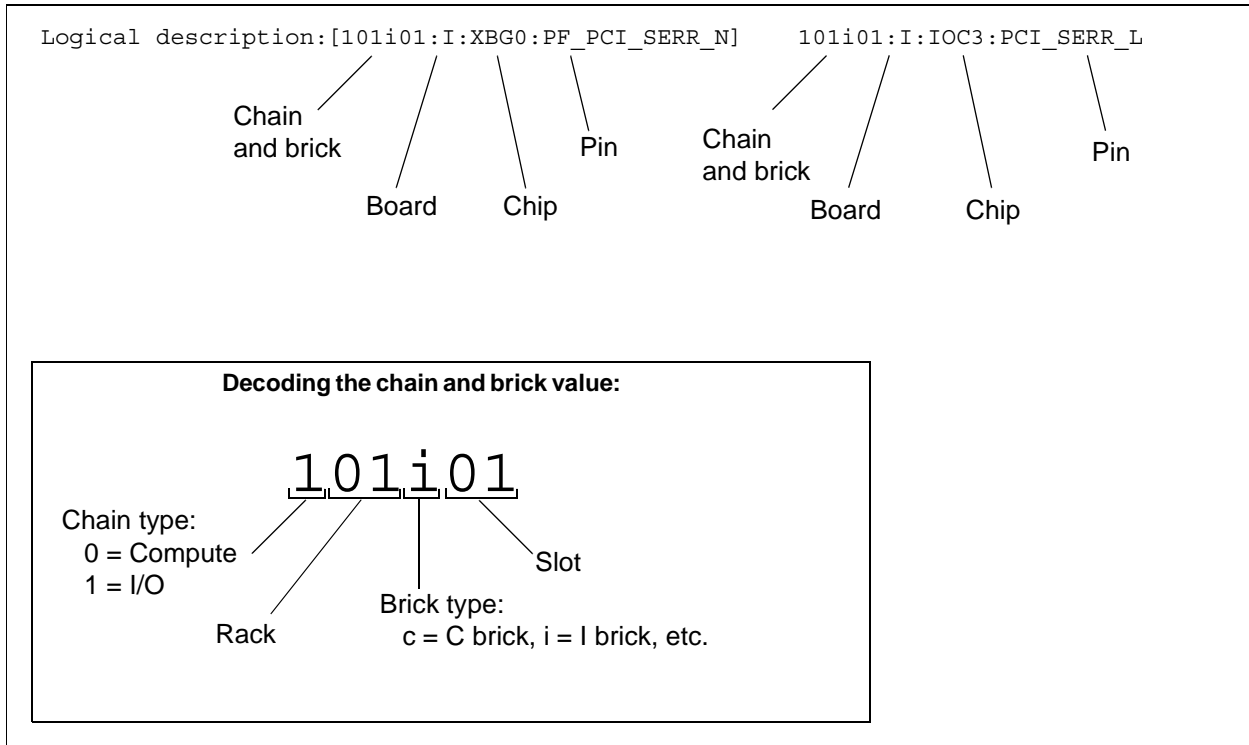


Figure 2-10 Logical Description Format

- Physical description: provides a physical description of the nodes on the failing net. The format of each node is <chain and brick>:<board>:<chip>:<pin>. scantest displays the node that detected the miscompare in square brackets [].

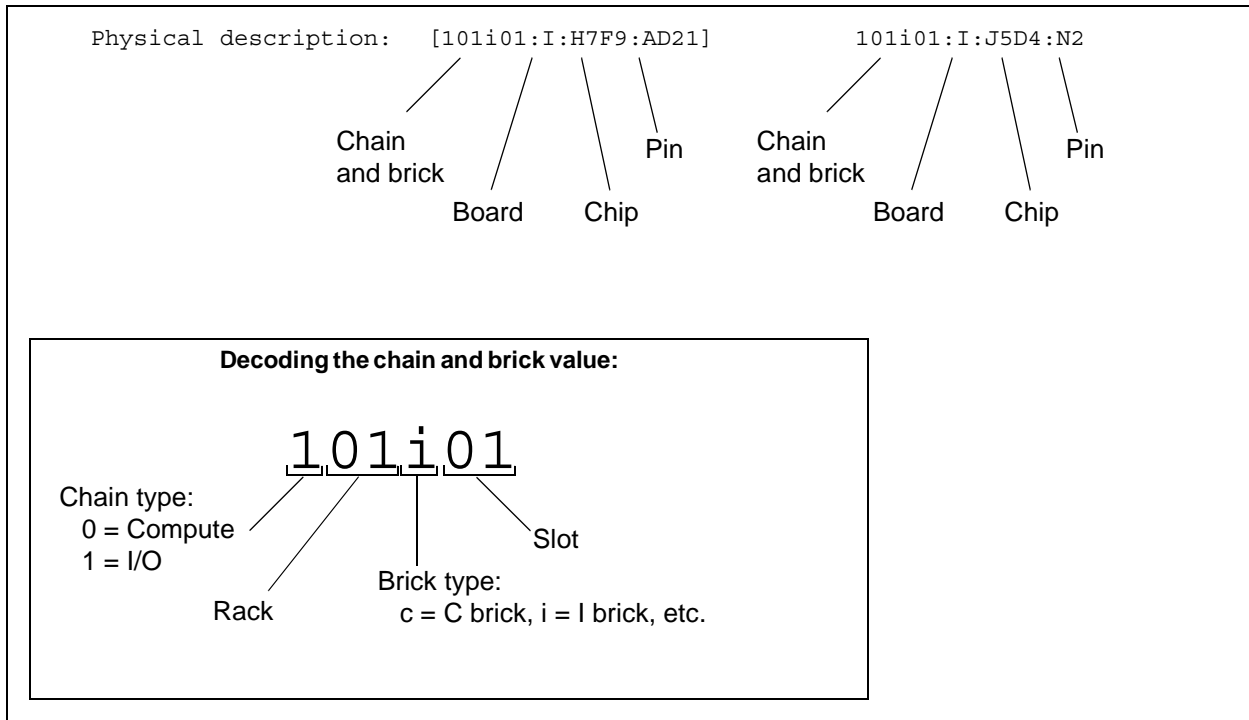


Figure 2-11 Physical Description Format

2.7 Physical and Logical Description to FRU Mapping

The easiest way to determine the failing FRU is by referring to the board failure information in the physical and logical descriptions. (Refer again to Figure 2-10 and Figure 2-11 to determine the location of the board information in the physical and logical descriptions.)

2.7.1 SGI Origin 300 Systems

Table 2-2 lists the FRUs that correspond to each board type that the boundary scan interconnect tests lists for SGI Origin 300 systems.

Table 2-2 FRUs Corresponding Board Type (SGI Origin 300 Systems)

Module	Board	FRU
Base module	NODE	IP45 4 CPU R14K 500MHZ 2MB IP45 2 CPU R14K 500MHZ 2MB
	DIMM0	DIMM 256MB 128MB DDR
	DIMM1	DIMM 512MB 128MB DDR (STANDARD)
	DIMM2	DIMM 1GB 256MB DDR
	DIMM3	
	I	CARD CAGE ASSY IO8 AND PCI
PCI expansion module	PCI1	PCA IO PCI 12 SLOT
	PCI2	

2.7.2 SGI 3000 Series Systems

Table 2-3 lists the FRUs that correspond to each board type that the boundary scan interconnect test lists for SGI 3000 series systems.

Table 2-3 FRUs Corresponding Board Type (SGI 3000 Series Systems)

Brick	Board	FRU
C	NODE	PCA IP35 4CPU TREX (P1)
	PIMM0	PCA IP35 PIMM R12KS 8MB 400MHZ PCA IP35 PIMM R12KS 8MB 400MHZ
	PIMM1	PCA IP35 PIMM R12KS 8MB 400MHZ PCA IP35 PIMM R12KS 8MB 400MHZ
	DIMM0	PCA 256MB DIMM 128MB DDR PCA 512MB DIMM 128MB DDR PCA 512MB DIMM 128MB DDR (Premium) PCA 1GB DIMM 256MB DDR
	DIMM1	PCA 256MB DIMM 128MB DDR PCA 512MB DIMM 128MB DDR PCA 512MB DIMM 128MB DDR (Premium) PCA 1GB DIMM 256MB DDR
	DIMM2	PCA 256MB DIMM 128MB DDR PCA 512MB DIMM 128MB DDR PCA 512MB DIMM 128MB DDR (Premium) PCA 1GB DIMM 256MB DDR
	DIMM3	PCA 256MB DIMM 128MB DDR PCA 512MB DIMM 128MB DDR PCA 512MB DIMM 128MB DDR (Premium) PCA 1GB DIMM 256MB DDR
	DIMM4	PCA 256MB DIMM 128MB DDR PCA 512MB DIMM 128MB DDR PCA 512MB DIMM 128MB DDR (Premium) PCA 1GB DIMM 256MB DDR
	DIMM5	PCA 256MB DIMM 128MB DDR PCA 512MB DIMM 128MB DDR PCA 512MB DIMM 128MB DDR (Premium) PCA 1GB DIMM 256MB DDR

Table 2-3 (continued) FRUs Corresponding Board Type (SGI 3000 Series Systems)

Brick	Board	FRU
C (cont.)	DIMM6	PCA 256MB DIMM 128MB DDR
		PCA 512MB DIMM 128MB DDR
		PCA 512MB DIMM 128MB DDR (Premium)
		PCA 1GB DIMM 256MB DDR
	DIMM7	PCA 256MB DIMM 128MB DDR
		PCA 512MB DIMM 128MB DDR
		PCA 512MB DIMM 128MB DDR (Premium)
		PCA 1GB DIMM 256MB DDR
I	I	PCA IO GENERAL P1
R	R	PCA SN1 ROUTER
P	P	PCA IO PCI 12 SLOT
X	X	PCA MIDPLANE XIO
G	GE0	PCA GE14-2
	GE1	PCA GE14-4
		PCA GE16-4
	KTOWN2	PCA ASSY KTOWN2
N	N0	PCA NULL BOARD
	N1	
	N2	
	N3	

2.8 Troubleshooting Tips

- If the boundary scan interconnect test detects an error during the SIC selection process, then a failure occurred in the scan hardware and/or the path to the SIC. Check the scan signals (TCK, TMS, TDI, TDO, and TRST) that go to the SIC and the SIC address lines.
- If the boundary scan interconnect test detects an error during the register level tests, then a failure occurred in the scan hardware and/or path after the SIC. Check the scan signals (TCK, TMS, TDI, TDO, and TRST) at each component in the chain.
- If the boundary scan interconnect test detects a failure during the interconnect test, identify and correct it before you run the scan-based link-level protocol interconnect test.

2.9 Example 1: Possible PIMM Failure

The following output shows a possible PIMM failure in an SGI 3000 series system:

```
Boundary-Scan Test Software (scantest V3.11) Mon Apr  2 11:31:20 2001

Loading chain configuration and reference data
  File: /stand/sysco/cfg/scan/c00155.cfg

Loading reference data for each board definition
  Path: /stand/sysco/data/scan/cmp

Initializing JTAG communications to scan controller(s)
  No hardware; running in software simulation/emulation mode

Initializing chain TAPs and configuring SICs:
001c01 with SIC at address 0x00 (CER=0x07)

Testing Integrity of Boundary Scan Instruction Registers
Performing IR length test on active UUTs:
001c01 SIC 0x0 ... passed (chain length=129)

Performing IR path test on active UUTs:
001c01 SIC 0x0 ... passed

Testing Integrity of Boundary Scan Bypass Registers
Performing BYPASS length test on selected UUTs:
001c01 SIC 0x0 ... passed (chain length=41)

Performing BYPASS path test on selected UUTs:
001c01 SIC 0x0 ... passed

Testing Integrity of Boundary Scan Data Registers
Performing chain-level BOUNDARY length test on selected UUTs:
001c01 SIC 0x0 ... passed (chain length=4226)

Performing chain-level BOUNDARY path test on selected UUTs:
001c01 SIC 0x0 ... passed

Performing system-level interconnect test on given configuration:
(22 of 22 patterns will be applied)
+++++
Errors were detected during interconnect test.

Failure information for BSR interconnect test:

Module and raw position : 000-001280
Expected data           : 1010100101011010010101
Actual data             : 0000000000000000000000
Difference data         : ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^
Driving node index      : 1111111111111111002211
Logical description     : 001c01:NODE:BR:SYSAD_P0_34_BUS
[001c01:PIMM0:CPUA:SYSAD_34_BUS] 001c01:PIMM0:CPUB:SYSAD_34_BUS
Physical description    : 001c01:NODE:J0J2:AL14 [001c01:PIMM0:F9C3:C12]
001c01:PIMM0:COC3:C12
Module and raw position : 000-001283
Expected data           : 1010011010100101010101
```

```
Actual data          : 111111111111111111111111
Difference data     :   ^ ^ ^   ^ ^ ^ ^ ^ ^ ^ ^
Driving node index  : 1111111111111111002211
Logical description : 001c01:NODE:BR:SYSAD_P0_40_BUS
[001c01:PIMM0:CPUA:SYSAD_40_BUS] 001c01:PIMM0:CPUB:SYSAD_40_BUS
Physical description : 001c01:NODE:J0J2:AR13 [001c01:PIMM0:F9C3:C16]
001c01:PIMM0:COC3:C16
```

Number of failing signals displayed from test: 2

Normal Program Termination

In this example, the boundary scan interconnect test detected two failing signals in the C brick located at 001c01.

The first signal is a connection between the Node (Bedrock ASIC) and PIMM0.

```
Logical description : 001c01:NODE:BR:SYSAD_P0_34_BUS
[001c01:PIMM0:CPUA:SYSAD_34_BUS] 001c01:PIMM0:CPUB:SYSAD_34_BUS
Physical description : 001c01:NODE:J0J2:AL14 [001c01:PIMM0:F9C3:C12]
001c01:PIMM0:COC3:C12
```

The second signal is also connection between the Node (Bedrock ASIC) and PIMM0.

```
Logical description : 001c01:NODE:BR:SYSAD_P0_40_BUS
[001c01:PIMM0:CPUA:SYSAD_40_BUS] 001c01:PIMM0:CPUB:SYSAD_40_BUS
Physical description : 001c01:NODE:J0J2:AR13 [001c01:PIMM0:F9C3:C16]
001c01:PIMM0:COC3:C16
```

The failing hardware could be PIMM0, the Bedrock ASIC, the board that contains them, or a connector between them. For this example, you would replace PIMM0 and run the test again.

2.10 Example 2: Possible DIMM Failure

The following output shows a possible DIMM failure in an SGI 3000 series system:

```
Boundary-Scan Test Software (scantest V3.11) Mon Apr  2 11:37:41 2001

Loading chain configuration and reference data
  File: /stand/sysco/cfg/scan/c001dimm04.cfg

Loading reference data for each board definition
  Path: /stand/sysco/data/scan/cmp

Initializing JTAG communications to scan controller(s)
  No hardware; running in software simulation/emulation mode

Initializing chain TAPs and configuring SICs:
001c01 with SIC at address 0x00 (CER=0x0C)

Testing Integrity of Boundary Scan Instruction Registers
Performing IR length test on active UUTs:
001c01 SIC 0x0 ... passed (chain length=149)

Performing IR path test on active UUTs:
001c01 SIC 0x0 ... passed

Testing Integrity of Boundary Scan Bypass Registers
Performing BYPASS length test on selected UUTs:
001c01 SIC 0x0 ... passed (chain length=19)

Performing BYPASS path test on selected UUTs:
001c01 SIC 0x0 ... passed

Testing Integrity of Boundary Scan Data Registers
Performing chain-level BOUNDARY length test on selected UUTs:
001c01 SIC 0x0 ... passed (chain length=1706)

Performing chain-level BOUNDARY path test on selected UUTs:
001c01 SIC 0x0 ... passed

Performing system-level interconnect test on given configuration:
(22 of 22 patterns will be applied)
+++++
Errors were detected during interconnect test.

Failure information for BSR interconnect test:

Module and raw position : 000-000022
Expected data           : 1010101010010101010101
Actual data             : 0000000000000000000000
Difference data         : ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^
Driving node index      : 00000000000000000001100
Logical description     : [001c01:NODE:BR:MB_DATA_43_BUS] 001c01:DIMM0:U2:A4
Physical description    : [001c01:NODE:J0J2:AD35] 001c01:DIMM0:A9B3:1
```

```
Module and raw position : 000-000643
Expected data          : 1001010110100101010101
Actual data           : 1111111111111111111111
Difference data        :  ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^
Driving node index     : 00000000000000000001100
Logical description    : [001c01:NODE:BR:MB_DATA_90_BUS] 001c01:DIMM4:U2:A3
Physical description   : [001c01:NODE:J0J2:AG05] 001c01:DIMM4:A9B3:64
```

Number of failing signals displayed from test: 2

Normal Program Termination

In this example, the boundary scan interconnect test detected two failing signals in the C brick located at 001c01.

The first signal is a connection between the Node (Bedrock ASIC) and DIMM0:

```
Logical description    : [001c01:NODE:BR:MB_DATA_43_BUS] 001c01:DIMM0:U2:A4
Physical description   : [001c01:NODE:J0J2:AD35] 001c01:DIMM0:A9B3:1
```

The second signal is a connection between the Node (Bedrock ASIC) and DIMM4:

```
Logical description    : [001c01:NODE:BR:MB_DATA_90_BUS] 001c01:DIMM4:U2:A3
Physical description   : [001c01:NODE:J0J2:AG05] 001c01:DIMM4:A9B3:64
```

The failing hardware could be the DIMMs, the Bedrock ASIC, the board that contains them, or a connector between them. For this example, you would swap DIMM0 and DIMM4 with other DIMMs that do not have failures to determine whether the failures move with the DIMM(s). If the failures move with the DIMM(s), you would replace the DIMM(s).

Scan-based Link-level Protocol Interconnect Test

This chapter describes the scan-based link-level protocol test.

3.1 About the Scan-based Link-level Protocol Interconnect Test

Several ASICs contain an additional boundary scan register (LLP register) that can be used to send a single micropacket (160 bits in eight 20-bit, time-MUXed transfers) from one LLP port to another port. The scan-based link-level protocol (LLP) interconnect test includes an application (`slit`) that uses the LLP registers to exercise router links at-speed. It tests one or two ports per router at a time.

This test uses a minimal number of patterns to determine whether a port is functional. If it detects an error, it attempts to isolate the failure to a single net. If it detects a single failing net, it identifies all chips and pins on the net and displays the actual/expected data.

The following ASICs support the at-speed LLP test feature:

- Router (ports A, B, C, D, E, F, G, and H)
- Bedrock (ports NI and II)
- Xbridge (ports 8, 9, A, B, C, and D)

Note: Several scripts are available to run the boundary scan interconnect test and scan-based link-level protocol interconnect test. Refer to Chapter 4, “Scripts That Run `scantest` and `slit`,” for more information.

3.2 Test Algorithm

This test uses the following algorithm:

1. Load the chain configuration and reference data.
2. Load the LLP link configuration and selection data.
3. Initialize JTAG communication to the scan controller(s).
4. Initialize the chain TAPs and configure the SIC(s).
5. Perform the scan hardware integrity tests:
 - Perform the IR length test.
 - Perform the chain-level LLP REG0 length test.
 - Perform the chain-level LLP REG1 length test.
6. Perform the scan-based LLP interconnect test.
7. Print a program termination status message.

Caution: Each test step must pass before the next step can be run. If the integrity tests in Step 5 do not pass, the interconnect test does not run because the results would be invalid. Do not manually run the interconnect test until the integrity tests pass.

Figure 3-1 shows the output from each step of this algorithm.

```

pyro% /stand/sysco/bin/slit -f /stand/sysco/cfg/scan/p001.cfg -i
/stand/sysco/cfg/scan/p001.links

Scan-based LLP Interconnect Test (slit 2.6) Tue Jun 27 11:00:28 2000

1 Loading chain configuration and reference data
    File: /stand/sysco/cfg/scan/p001.cfg

2 Loading LLP link configuration and selection data
    File: /stand/sysco/cfg/scan/p000.links

3 Initializing JTAG communications to scan controller(s)

4 Initializing chain TAPs and configuring SICs:
    101p01 with SIC at address 0x0A (CER=0x08)

    Performing IR length test on active UUTs:
    101p01 ... passed (chain length=9)

5 Performing chain-level LLP REG0 length test on selected UUTs:
    101p01 ... passed (chain length=1011)

    Performing chain-level LLP REG1 length test on selected UUTs:
    101p01 ... passed (chain length=3)

6 Performing scan-based LLP interconnect test on selected link:

    Driving port (SSD) = P:U0:A
    Receiving port (SSR) = P:U0:A
    (number of patterns=4; flit mask per pattern=0xFF)
    ++++++*****+*****+*****+*****+*****+*****+*****+*****+*****
    Errors were detected during interconnect test.

    Failure information for scan-based LLP interconnect test:

    Pattern (8-bits each) : ---01--- ---02--- ---03--- ---04---
    Actual data           : 01101010 0000.... 0000.... 10000000
    Difference data       :      ^^      ^^ ^      ^^ ^      ^ ^ ^
    Logical description   : 101p01:U0:PORT_A_OUT_DATA_6
    [101p01:U0:PORT_A_IN_DATA_6]

7 Normal Program Termination

```

Figure 3-1 slit Test Algorithm Components

3.3 slit Command-line Options

`slit` uses the following command syntax:

```
slit [-h] <-f file> <-i file> [-ID num] [-d file] [-p num] [-e num] [-t test]
[-m mode] [-l level] [-C] [-s mode] [-L pats] [-F bit_mask] [-R] [-v] [--l2 <host>
| --interface <iface> | --ssn <serial>[:<rack>] | --sysname
<system_name>[:<rack>] | --scdev <device> | --serial <host>:<port> | --dev
<device_name>]
```

Table 3-1 describes the command-line options that you can use to control `slit`.

Table 3-1 slit Command-line Options

Option	Description
-h	Displays the available command-line options and then exits.
-f <file>	Specifies a configuration file to use. The configuration file defines the hardware to test. (Default: <code>slit.cfg</code>) Use the following configuration files to test entire systems: <code>sys8p.cfg</code> = system with 8 processors <code>sys16p.cfg</code> = system with 16 processors <code>sys32p.cfg</code> = system with 32 processors <code>sys64p.cfg</code> = system with 64 processors <code>sys128p.cfg</code> = system with 128 processors Use the following configuration files to target specific bricks: <code>c001.cfg</code> = IP35 C brick <code>i002.cfg</code> = I brick <code>p001.cfg</code> = P brick <code>r001.cfg</code> = R brick <code>x001.cfg</code> = X brick <code>n001.cfg</code> = N brick <code>g001.cfg</code> = G brick <code>ip45_001.cfg</code> = IP45 motherboard and CPUs (SGI Origin 300 system) Other configuration files are available in the <code>/stand/sysco/cfg/scan/</code> directory. Refer to the <code>/stand/sysco/cfg/scan/README.cfg</code> file for more information.

Table 3-1 (continued) slit Command-line Options

Option	Description
-i <file>	<p>Specifies a file that contains input for the test. The input file describes the port-to-port connections that will be tested. (Default: <code>slit.in</code>)</p> <p>Use the following files to test entire systems:</p> <pre> sys8p.links = system with 8 processors sys16p.links = system with 16 processors sys32p.links = system with 32 processors sys64p.links = system with 64 processors sys128p.links = system with 128 processors </pre> <p>Use the following files to target specific bricks:</p> <pre> c001.links = IP35 C brick i002.links = I brick p001.links = P brick r001.links = R brick x001.links = X brick n001.links = N brick ip45_001ni.links = SGI Origin 300 system </pre>
-ID <num>	<p>Overrides the ID field listed in a configuration file. This enables you select a different brick for testing without modifying the configuration file. Use the following formula to select a different brick:</p> <p>ID = (rack * 64) + slot</p> <p>For example, to test a P brick in slot 7 of rack 2, enter the following command:</p> <pre> slit -f /stand/sysco/cfg/scan/p001.cfg -ID 135 -i /stand/sysco/cfg/scan/p001.links </pre>
-d <file>	<p>Specifies a file to receive output from the test. (Default: <code>slit.dmp</code>)</p>
-p <num>	<p>Specifies the number of test passes that <code>slit</code> should perform. (Valid values are between 1 and 10,000; default: 1)</p>
-e <num>	<p>Specifies the maximum number of errors to display from the interconnect test. (Range: 0 to 10,000; default: 200)</p> <p>If you specify 0 errors, no detailed failure information is reported, which may be useful if the test is run from an automated application.</p>

Table 3-1 (continued) slit Command-line Options

Option	Description
-t <test>	<p>Specifies the tests to run.</p> <p>Valid tests:</p> <ul style="list-style-type: none">irp = instruction register path testirl = instruction register length testir = instruction register path test and instruction register length testprp = bypass register path testprl = bypass register length testpr = bypass register path test and bypass register length testlrp = LLP register path testlrl = LLP register length testlr = LLP path test and boundary length testllp = LLP at-speed interconnect testid = read and display device ID registersall = all tests <p>Default test selection: irl+lrl+llp</p>
-m <mode>	<p>Specifies the mode for the register tests.</p> <p>Valid modes:</p> <ul style="list-style-type: none">normal = normal lengths and minimal pattern(s)elen = extended lengthsepat = extended pattern set <p>If normal is specified, then no other option is valid. (Default: normal)</p>
-l <level>	<p>Specifies the level of the register tests.</p> <p>Valid modes:</p> <ul style="list-style-type: none">chn = chain level register testbrd = board level register testchip = chip level register test <p>(Default: chn)</p>
-C	<p>Selects an individual Scan Interface Chip (SIC) to run the test. (The test will not run on any other SICs.)</p> <p>When you use this option:</p> <ul style="list-style-type: none">You can specify one of the following tests with -t: irl, irp, ir, prl, prp, prThe -l option is ignored.The -m option is used for lengths and patterns.
-s <mode>	<p>Selects a stepping mode.</p> <p>Valid modes:</p> <ul style="list-style-type: none">no = no test stepping; run continuousall = step through each pattern; prompt usererr = step through failing pattern; prompt user <p>(Default: no)</p>

Table 3-1 (continued) slit Command-line Options

Option	Description
-L <pat>	Selects the LLP test pattern that the test uses. Valid options: detect = patterns to detect interconnect faults isolate = patterns to isolate short/bridging faults stress = patterns to stress the interconnects custom = user-defined patterns (Default: detect)
-F <mask>	Selects the LLP flits that the test should capture and compare. (Each test packet contains eight separate flits.) The mask parameter is a bitmap of the chosen flits: If bit 0 is set, then flit 0 is captured/compared. If bit 1 is set, then flit 1 is captured/compared. (This parameter can be a hexadecimal number between 0x01 and 0xFF; default: 0xFF)
-R	Disables use of resets.
-v	Selects verbose mode, which writes additional messages to the output.
--l2 <host>	Connects to the L2 system controller with the specified host name or IP address
--interface <iface>	Connects to the L2 system controller with the specified network interface (for example, eth0)
--ssn <serial> [: <rack>]	Connects to the system with the specified serial number (for example, L01234567)
--sysname <system_name> [: <rack>]	Connects to the system specified by <system_name>
--scdev <device>	Connects to the specified system controller
--serial <host> : <port>	Connects to the specified Ethernet host name and port
--dev <device_name>	Connects to the specified local serial port device

3.4 Running the Scan-based Link-level Protocol Interconnect Test

Perform the following procedure to run the scan-based link-level protocol interconnect test:

Warning: Ensure that the system is offline before you run this test. If the operating system is running when you use this test, the operating system will crash and customer data will be lost.

1. Power up the system.
2. Log into the L3 system controller as `sgidiag` (default password: `sgi!diag`).
3. Enter the `cd /stand/sysco/usr` command to change to the `/stand/sysco/usr` directory.
4. Connect to the L2 system controller.
 - To use a hardware L2 connection:
Connect the L3 system controller to the L2 system controller with an Ethernet connection by using a crossover cable.

Obtain the IP address of the L2 system controller:

```
cat /var/state/dhcp/dhcpd.leases
```


Enter the command: `/stand/sysco/l2term --l2 <host>`
 - To use the L2 emulator:
Enter the command: `/stand/sysco/bin/l2`
5. Enter the appropriate command to start the test:
 - For the hardware L2 connection:

```
/stand/sysco/bin/slit -f /stand/sysco/cfg/scan/<file> -i  
/stand/sysco/cfg/scan/<file> --l2 <host> [test_options]
```


Example: `/stand/sysco/bin/slit -f /stand/sysco/cfg/scan/sys8p.cfg
-i /stand/sysco/cfg/scan/sys8p.links --l2 10.1.1.37`
 - For the L2 emulator:

```
/stand/sysco/bin/slit -f /stand/sysco/cfg/scan/<file> -i  
/stand/sysco/cfg/scan/<file> [test_options]
```


Example: `slit -f /stand/sysco/cfg/scan/sys8p.cfg -i
/stand/sysco/cfg/scan/sys8p.links`

Refer again to Table 3-1 for descriptions of the available command-line options.
6. Interpret the output. (Refer to Section 3.5, “Scan-based Link-level Protocol Interconnect Test Output” for more information.)
7. When you are done testing, enter the L2 `reset` command to reset the system.

3.5 Scan-based Link-level Protocol Interconnect Test Output

When this test detects an error, it continues testing the entire suite of test vectors before it generates error output. Once it has completed the entire suite of test vectors, it generates output that shows the actual and expected values for a miscompare. It also displays the specific chip and pin information for each net that had a miscompare. (The failing chip and pin are identified by logical location.)

The error output uses the following format:

```
Errors were detected during interconnect test.
```

```
Failure information for scan-based LLP interconnect test:
```

```
Pattern (8-bits each) : ---01--- ---02--- ---03--- ---04---
Actual data           : 01101010 0000.... 0000.... 10000000
Difference data       :   ^^   ^^ ^   ^^ ^   ^ ^ ^
Logical description   : 101p01:U0:PORT_A_OUT_DATA_6
[101p01:U0:PORT_A_IN_DATA_6]
```

The error information contains the following information:

- **Pattern:** indicates the data pattern that `slit` was using when the miscompare occurred. (Each data signal transfers 8 bits of data per test pattern. The number of patterns applied is determined by the pattern type and LLP version. `slit` supports three predefined pattern sets [detect, isolate, and stress] and user-defined custom patterns.)
- **Actual data:** shows the actual data that was read from the system. (If valid data was not captured for a specific pattern [due to test configuration or a problem] a period (.) is shown.)
- **Difference data:** contains a ^ symbol for each pattern in the signature where the actual data did not match the expected data.
- **Logical description:** provides a logical description of the ports on the failing connection.

The format of each node is `<board>:<chip>:<pin>`. The driver is listed first, followed by the receiver; the receiver is the node that detected the miscompare.

The board information in the logical description indicates the FRU to replace (refer again to Table 2-2 and Table 2-3).

The format of the `<pin>` is `PORT_<port>_<dir>_DATA_<num>`, in which:

`<port>` = LLP port on ASIC (for Xbridge: 8, 9, A, B, C, D)

`<dir>` = OUT for SSD output; IN for SSR input

`<num>` = Data line number (0 - 19)

Note: The test can only identify a differential pair; it cannot isolate which signal of the pair is failing.

3.6 Troubleshooting Tips

- If the fault is detected between the A and B ports, use a *single-wrap* cable or loopback cable on each port to isolate the errors to a specific port. Replace the HIC on a failing port to isolate the error to the motherboard or HIC.
- If multiple failures are detected on the A or B ports, double-check the seating of the HIC and cable.
- If an entire port fails (20 data lines, 2 clock lines and data ready) and all of the inputs are sampling zero (0), check the Widget Present signal. (The receivers are disabled if the Widget Present is not correct.)
- If the `slit` LLP register test fails but the boundary scan interconnect test passes, check the system clock (required for LLP register to shift).
- If the LLP-register interconnect test does not capture valid data for any pattern, use the L1 system controller to verify that both ports are configured with the same speed. Then, verify that the clocks are running.

3.7 Example

For this example, assume that `slit` generates the following error information:

```
SCAN-BASED LLP INTERCONNECT FAILURE
-----

Pattern (8-bits each) : ---01--- ---02--- ---03--- ---04---
Actual data           : 00000000 00000000 00000000 00000000
Difference data       : ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^
Logical description   : 101p01:P:XBG0:PORT_9_OUT_DATA_6
[101p01:P:XBG1:PORT_8_IN_DATA_6]
```

From the output in this example, you can determine the following information:

Node Number	Chain/ Brick	Board Name	Logical Chip	Logical Pin	LLP Port	Signal Type	Data Line
0	101p01	P	XBG0	PORT_9_OUT_DATA_6	9 SSD	OUT	6
1	101p01	P	XBG1	PORT_8_IN_DATA_6	8 SSR	IN	6

In the P brick at location 101p01, the data is driven from the port 9 SSD on XBG0 and received in the port 8 SSR on XBG1. The data line is valid for all bits of each pattern, and captured all zeroes (0's).

The board information in the logical description indicates the FRU to replace (refer again to Table 2-2 and Table 2-3). For this example, the board information indicates that the P brick located at 101p01 is the FRU to replace.

Scripts That Run scantest and slit

This chapter describes the `scan_speedo2`, `scan_pebrick`, `scan_nbrick`, and `scan_kcar_11` scripts that run the `scantest` and `slit` tests on SGI Origin 300 systems, PCI expansion modules, N bricks, and G bricks.

4.1 scan_speedo2 Script

The `scan_speedo2` script runs the `scantest` and `slit` tests on an SGI Origin 300 system. `scan_speedo2` uses the following syntax:

```
scan_speedo2 [-p] [-h]
```

Table 4-1 describes the command-line options that you can use to control `scan_speedo2`.

Table 4-1 scan_speedo2 Command-line Options

Option	Description
-p	Disables the script from prompting you to press <Enter> before it starts each test
-h	Displays help information

Perform the following procedure to run this script:

1. Power up the system.
2. Log into the L3 system controller as `sgidiag` (default password: `sgi!diag`).
3. Enter `cd /stand/sysco/usr` to change to the `/stand/sysco/usr` directory.
4. Enter `/stand/sysco/bin/12` to start the L2 emulator.
5. Configure the system as rack 1, slot 1.
6. Enter `/stand/sysco/bin/scan_speedo2` to start the script.
7. Enter `1` to run the test on an IP45 system.
8. Enter `2` or `4` to specify that your system has 2 or 4 processors.
9. Select the tests that you want to run.

Table 4-2 describes the tests that are valid for field use. (Test options 4, 6, and 9 require special hardware that is not available in the field.)

Table 4-2 scan_speedo2 Test Options

Option	Description
1	Tests the processor cache by checking the interconnects between the CPUs and cache SRAMs
2	Tests the interconnects between the CPUs, XBridge ASIC, and Bedrock ASIC
3	Tests the interconnects between NI and II ports (requires a loopback cable between the NI and II ports)
5	Tests the interconnects between the IP45 and IO8 boards and checks the interconnects on the IO8 board
7	Tests the on-board links and the loopback connections between the NI and IO ports (requires a loopback cable between the NI and IO ports)
8	Runs all of the tests that do not require additional loopback cables (options 1, 2, and 5)

10. Interpret the output. The script returns output to the screen and to the following files in the directory from which you started the script:

- ip45_001_sc.dmp (scantest output from test option 1)
- ip45_002.dmp (scantest output from test option 2)
- ip45_002ni.dmp (scantest output from test option 3)
- speedo2_008.dmp (scantest output from test option 5)
- slit.dmp (slit output from test option 7)
- all_tests.dmp (output from all tests)

Refer to Chapter 2, “Boundary Scan Interconnect Test,” and Chapter 3, “Scan-based Link-level Protocol Interconnect Test,” for more information about interpreting the output from `scantest` and `slit`.

Example:

```
/stand/sysco/usr> /stand/sysco/bin/scan_speedo2
```

```
--- SCAN INTERCONNECT TEST OF SPEEDO2 (scan_speedo2 version 1.1) ---
```

Run scan interconnect tests on SpeedoII brick with IP39 node board.
Brick location MUST be set to rack=1, slot=1 during scan test.

The tests support two Node board types: IP39 or IP45
Select from the following types:

- 0) IP39
 - 1) IP45
- > 1

The tests support two or four processors on the IP45.
Select from the following processor configuration for the IP45:

- 2) 2 CPUs (CPUA and CPU1A only)
 - 4) 4 CPUs (CPUA, CPUB, CPU1A, CPU1B)
- > 4

There are several interconnect tests with different areas of coverage.
For complete test of an Speedo2 brick, install scannable four DIMMs,
two scannable PCIs and loopback cables; run all available tests.

Select from the following interconnect tests available:

- 1) IP45 processor cache (nets between CPUs and SRAMs)
 - 2) IP45 system interface (nets between CPUs, XBRIDGE and BEDROCK)
 - 3) IP45 system interface with loopbacks on NI and IO ports
 - 4) IP45 main memory (nets between CPUs and scannable DIMMs)
 - 5) Speedo2 brick (nets between IP45 and IO8 boards, and nets on IO8)
 - 6) Speedo2 brick PCI slots (nets between IP45 and PCI slots)
 - 7) IP45 at-speed LLP test (on-board link and loopbacks on NI and IO)
-
- 8) All tests for Speedo2 brick stand-alone (1, 2, 5)
 - 9) All tests for Speedo2 with SDTBs, SPTBs and loopbacks (1-7)
- > 1

The L2 software emulator must be running if real L2 not used.
Turn on power to boards under test (IP45/IO8/PCI).
Brick location MUST be set to rack=1, slot=1 during scan test.

```

--- TEST 1: IP45 processor cache ---
Press ENTER when ready to start running selected test...
<Enter>

Boundary-Scan Test Software (scantest V3.12) Thu Sep  6 15:54:09 2001

Loading chain configuration and reference data
  File: /stand/sysco/cfg/scan/ip45_001_sc.cfg

Loading reference data for each board definition
  Path: /stand/sysco/data/scan/cmp

Initializing JTAG communications to scan controller(s)
  No hardware; running in software simulation/emulation mode

Initializing chain TAPs and configuring SICs:
  001c01 with SIC at address 0x01 (CER=0x03)

Testing Integrity of Boundary Scan Instruction Registers
Performing IR length test on active UUTs:
  001c01 SIC 0x1 ... passed (chain length=76)

Performing IR path test on active UUTs:
  001c01 SIC 0x1 ... passed

Testing Integrity of Boundary Scan Bypass Registers
Performing BYPASS length test on selected UUTs:
  001c01 SIC 0x1 ... passed (chain length=24)

Performing BYPASS path test on selected UUTs:
  001c01 SIC 0x1 ... passed

Testing Integrity of Boundary Scan Data Registers
Performing chain-level BOUNDARY length test on selected UUTs:
  001c01 SIC 0x1 ... passed (chain length=2832)

Performing chain-level BOUNDARY path test on selected UUTs:
  001c01 SIC 0x1 ... passed

Performing board-level interconnect test on selected boards:

  001c01 NODE (#Patterns=22)
  ++++++
  No errors detected during interconnect test.

Normal Program Termination
Results stored in log file: ip45_001_sc.dmp

NOTE: Check results for each of the individual tests run.
      Results for all tests stored in log file: all_tests.dmp

```

4.2 scan_pebrick Script

The `scan_pebrick` runs the `scantest` and `slit` tests on a PCI expansion module connected to an SGI Origin 300 system. `scan_pebrick` uses the following syntax:

```
scan_pebrick [-p] [-h] [-m <id>] [-l <l2>]
```

Table 4-3 describes the command-line options that are available.

Table 4-3 scan_pebrick Command-line Options

Option	Description
-p	Disables the script from prompting you to press <Enter> before it starts each test
-h	Displays help information
-m <id>	Overrides the ID field listed in a configuration file. This enables you select a different brick for testing without modifying the configuration file. Use the following formula to select a different brick: ID = (rack * 64) + slot
-l <l2>	Specifies the IP address of the L2 system controller (software or hardware)

Perform the following procedure to run this script:

1. Power up the system.
2. Log into the L3 system controller as `sgidiag` (default password: `sgi!diag`).
3. Enter the `cd /stand/sysco/usr` command to change to the `/stand/sysco/usr` directory.
4. Connect to the L2 system controller.
 - To use a hardware L2 connection:

Connect the L3 system controller to the L2 system controller with an Ethernet connection via a crossover cable.

Obtain the IP address of the L2 system controller:
`cat /var/state/dhcp/dhcpd.leases`

Enter the command: `/stand/sysco/l2term --l2 <host>`

Example: `/stand/sysco/l2term --l2 10.1.1.37`
 - To use the L2 emulator:

Enter the command: `/stand/sysco/bin/l2`
5. Configure the SGI Origin 300 base module that is connected to the PCI expansion module as rack 1, slot 2.
6. Configure the PCI expansion module (PE brick) as rack 1, slot 2.
7. Enter `/stand/sysco/bin/scan_pebrick` to start the script.

Table 4-4 describes the tests that are valid for field use. (Test options 3, 4, 5, 6, 7, and 8 require special hardware that is not available in the field.)

Table 4-4 scan_pebrick Test Options

Option	Description
1	Tests the interconnects between Xbridge 0, 1, and 2 on the PE brick motherboard
2	Tests the interconnects on the PE brick motherboard and port A or B loopback
9	Performs at-speed test of the PE brick motherboard
10	Tests the interconnects on the PE brick motherboard and BIC 1
11	Tests the interconnects on the PE brick motherboard and BIC 2
12	Tests the connection between the SGI Origin 300 system and the PE brick
13	Performs at-speed test of the PE brick motherboard and port A/B loopback
14	Performs at-speed test of the PE brick motherboard and BIC 2
15	Performs at-speed test of the PE brick motherboard and BIC 2
16	Performs at-speed test of the connection between the SGI Origin 300 system and the PE brick

8. Interpret the output. The script returns output to the screen and to the following files in the directory from which you started the script:

- p001.dmp (scantest output from test option 1)
- p001_b.dmp (scantest output from test option 2)
- p001.dmp (slit output from test option 9)
- pe001_b1.dmp (scantest output from test option 10)
- pe001_b2.dmp (scantest output from test option 11)
- pe001_S2B1.dmp (scantest output from test option 12)
- p001_b.dmp (slit output from test option 13)
- pe001_b1.dmp (slit output from test option 14)
- pe001_b2.dmp (slit output from test option 15)
- pe001_S2B1.dmp (slit output from test option 16)
- pe001_C2PE_B.dmp (scantest output from test option 17)
- p001_a.dmp (slit output from test option 18)
- pe001_S2B2.dmp (slit output from test option 19)
- all_tests.dmp (output from all tests)

Refer to Chapter 2, “Boundary Scan Interconnect Test,” and Chapter 3, “Scan-based Link-level Protocol Interconnect Test,” for more information about interpreting the output from `scantest` and `slit`.

Example:

```
/stand/sysco/usr> /stand/sysco/bin/scan_pebrick  
--- SCAN INTERCONNECT TEST OF PE-Brick (scan_pebrick version 1.0) ---
```

Run scan interconnect tests on PE-Brick.

To run interconnect test between a PE-Brick and an Speedo2
or a C-Brick the PE-Brick id must be 001p02 and the Speedo2
or C-Brick id msut be 001c01.

The tests support two configurations.

Select from the following configurations:

0) Connected to Speedo2 via BIC

1) Connected to C-Brick via HIC

> 0

Enter the HIC/BIC port number connected to the C-Brick/Speedo2

1) HIC/BIC port A

2) HIC/BIC port B

> 1

There are several interconnect tests with different areas of coverage.

Select from the following interconnect tests available:

1) PE Brick Motherboard (nets between xbridges 0, 1, 2)

2) PE Brick Motherboard and Port B Loopback

3) PE Brick Motherboard and PCI slots Bus 1

4) PE Brick Motherboard and PCI slots Bus 2

5) PE Brick Motherboard and PCI slots Bus 3

6) PE Brick Motherboard and PCI slots Bus 4

7) PE Brick Motherboard and PCI slots Bus 5

8) PE Brick Motherboard and PCI slots Bus 6

9) At speed (LLP) PE Brick Motherboard

10) PE Brick Motherboard and BIC 1

11) PE Brick Motherboard and BIC 2

12) Link between Speedo2 and PE-Brick

13) At speed (LLP) PE Brick Motherboard and Port B Loopback

14) At speed (LLP) PE Brick Motherboard and BIC 1

15) At speed (LLP) PE Brick Motherboard and BIC 2

16) At speed (LLP) link between Speedo2 and PE-Brick

> 1

The L2 software emulator must be running if real L2 not used.

Turn on power to boards under test.

Brick location MUST be set to rack=1, slot=1 during scan test

unless the -m option was used.

--- TEST 1: PE Brick Motherboard ---

Press ENTER when ready to start running selected test...

<Enter>

Boundary-Scan Test Software (scantest V3.13) Tue Jan 15 10:49:47 2002

Loading chain configuration and reference data

File: /stand/sysco/cfg/scan/p001.cfg

Loading reference data for each board definition

Path: /stand/sysco/data/scan/cmp

Initializing JTAG communications to scan controller(s)

No hardware; running in software simulation/emulation mode

Initializing chain TAPs and configuring SICs:

101p01 with SIC at address 0x0A (CER=0x08)

Testing Integrity of Boundary Scan Instruction Registers

Performing IR length test on active UUTs:

101p01 SIC 0xA ... passed (chain length=9)

Performing IR path test on active UUTs:

101p01 SIC 0xA ... passed

Testing Integrity of Boundary Scan Bypass Registers

Performing BYPASS length test on selected UUTs:

101p01 SIC 0xA ... passed (chain length=3)

Performing BYPASS path test on selected UUTs:

101p01 SIC 0xA ... passed

Testing Integrity of Boundary Scan Data Registers

Performing chain-level BOUNDARY length test on selected UUTs:

101p01 SIC 0xA ... passed (chain length=1881)

Performing chain-level BOUNDARY path test on selected UUTs:

101p01 SIC 0xA ... passed

Performing board-level interconnect test on selected boards:

101p01 P (#Patterns=18)

+++++

No errors detected during interconnect test.

Normal Program Termination

Results stored in log file: p001.dmp

NOTE: Check results for each of the individual tests run.

Results for all tests stored in log file: all_tests.dmp

4.3 scan_nbrick Script

The `scan_nbrick` script runs the `scantest` and `slit` tests on an N brick in an SGI 3000 series system. `scan_nbrick` uses the following syntax:

```
scan_nbrick
```

Perform the following procedure to run this script:

1. Power up the system.
2. Log into the L3 system controller as `sgidiag` (default password: `sgi!diag`).
3. Enter the `cd /stand/sysco/usr` command to change to the `/stand/sysco/usr` directory.
4. Connect to the L2 system controller.
 - To use a hardware L2 connection:
Connect the L3 system controller to the L2 system controller with an Ethernet connection via a crossover cable.

Obtain the IP address of the L2 system controller:

```
cat /var/state/dhcp/dhcpd.leases
```


Enter the command: `/stand/sysco/l2term --l2 <host>`
 - To use the L2 emulator:
Enter the command: `/stand/sysco/bin/l2`
5. Enter `/stand/sysco/bin/scan_nbrick` to start the script.
6. Enter `0` to select C-Brick (IP35 only).
7. Enter `1` or `4` to select the number of null boards in the N brick.
 - If you entered `1`, enter the number of the slot that contains the null board to test.
 - If you entered `4`, select the appropriate cabling method (`0` for A-to-A and B-to-B loopback or `1` for A-to-B cabling).
8. Interpret the output.

4.3.1 Example 1: Testing Four Null Boards in an N Brick

```
/stand/sysco/usr> /stand/sysco/bin/scan_nbrick
--- SCAN INTERCONNECT TESTS OF N-Brick

Select from the following N brick communication methods
0) C-Brick (IP35 only)
1) Power Up Box
-1) Exit
> 0

Specify the number of NULL boards installed in the N Brick
1 or 4
-1 exit
> 4

Enter the cabling option
0) A<->A and B<->B loopback
1) A<->B cable
-1 exit
> 0

Test of N Brick with 4 Null Boards installed.
Insure all cables are connected and power up the brick
Press return when ready to run test
> <Enter>
Config name: [n001_CF.cfg]

Boundary-Scan Test Software (scantest V3.13) Tue Jan 15 10:49:00 2002

Loading chain configuration and reference data
  File: /stand/sysco/cfg/scan/n001_CF.cfg

Loading reference data for each board definition
  Path: /stand/sysco/data/scan/cmp

Initializing JTAG communications to scan controller(s)
  No hardware; running in software simulation/emulation mode

Initializing chain TAPs and configuring SICs:
001c01 with SIC at address 0x00 (CER=0x04)
001n02 with SIC at address 0x00 (CER=0x01)
001n02 with SIC at address 0x01 (CER=0x01)
001n02 with SIC at address 0x02 (CER=0x01)
001n02 with SIC at address 0x03 (CER=0x01)

Testing Integrity of Boundary Scan Instruction Registers
Performing IR length test on active UUTs:
001c01 SIC 0x0 ... passed (chain length=5)
001n02 SIC 0x0 ... passed (chain length=3)
001n02 SIC 0x1 ... passed (chain length=3)
001n02 SIC 0x2 ... passed (chain length=3)
001n02 SIC 0x3 ... passed (chain length=3)
```

Performing IR path test on active UUTs:

001c01 SIC 0x0 ... passed
001n02 SIC 0x0 ... passed
001n02 SIC 0x1 ... passed
001n02 SIC 0x2 ... passed
001n02 SIC 0x3 ... passed

Testing Integrity of Boundary Scan Bypass Registers

Performing BYPASS length test on selected UUTs:

001c01 SIC 0x0 ... passed (chain length=1)
001n02 SIC 0x0 ... passed (chain length=1)
001n02 SIC 0x1 ... passed (chain length=1)
001n02 SIC 0x2 ... passed (chain length=1)
001n02 SIC 0x3 ... passed (chain length=1)

Performing BYPASS path test on selected UUTs:

001c01 SIC 0x0 ... passed
001n02 SIC 0x0 ... passed
001n02 SIC 0x1 ... passed
001n02 SIC 0x2 ... passed
001n02 SIC 0x3 ... passed

Testing Integrity of Boundary Scan Data Registers

Performing chain-level BOUNDARY length test on selected UUTs:

001c01 SIC 0x0 ... passed (chain length=842)
001n02 SIC 0x0 ... passed (chain length=627)
001n02 SIC 0x1 ... passed (chain length=627)
001n02 SIC 0x2 ... passed (chain length=627)
001n02 SIC 0x3 ... passed (chain length=627)

Performing chain-level BOUNDARY path test on selected UUTs:

001c01 SIC 0x0 ... passed
001n02 SIC 0x0 ... passed
001n02 SIC 0x1 ... passed
001n02 SIC 0x2 ... passed
001n02 SIC 0x3 ... passed

Performing system-level interconnect test on given configuration:

(18 of 18 patterns will be applied)

+++++

No errors detected during interconnect test.

Normal Program Termination

4.3.2 Example 2: Testing One Null Board in an N Brick

```
/stand/sysco/usr> /stand/sysco/bin/scan_nbrick

--- SCAN INTERCONNECT TESTS OF N-Brick

Select from the following N brick communication methods
0) C-Brick (IP35 only)
1) Power Up Box
-1) Exit
> 0

Specify the number of NULL boards installed in the N Brick
1 or 4
-1 exit
> 1

Enter the slot number for the NULL board
0 to 3
-1 exit
> 0

Test of N Brick with 1 Null Board installed in slot 0 .
Insure all cables are connected and power up the brick
Press return when ready to run test
> <Enter>
Config name: [n001_C1.cfg]

Boundary-Scan Test Software (scantest V3.13) Tue Jan 15 10:49:12 2002

Loading chain configuration and reference data
  File: /stand/sysco/cfg/scan/n001_C1.cfg

Loading reference data for each board definition
  Path: /stand/sysco/data/scan/cmp

Initializing JTAG communications to scan controller(s)
  No hardware; running in software simulation/emulation mode

Initializing chain TAPs and configuring SICs:
001c01 with SIC at address 0x00 (CER=0x04)
001n02 with SIC at address 0x00 (CER=0x01)

Testing Integrity of Boundary Scan Instruction Registers
Performing IR length test on active UUTs:
001c01 SIC 0x0 ... passed (chain length=5)
001n02 SIC 0x0 ... passed (chain length=3)

Performing IR path test on active UUTs:
001c01 SIC 0x0 ... passed
001n02 SIC 0x0 ... passed

Testing Integrity of Boundary Scan Bypass Registers
Performing BYPASS length test on selected UUTs:
001c01 SIC 0x0 ... passed (chain length=1)
001n02 SIC 0x0 ... passed (chain length=1)
```

Performing BYPASS path test on selected UUTs:

001c01 SIC 0x0 ... passed

001n02 SIC 0x0 ... passed

Testing Integrity of Boundary Scan Data Registers

Performing chain-level BOUNDARY length test on selected UUTs:

001c01 SIC 0x0 ... passed (chain length=842)

001n02 SIC 0x0 ... passed (chain length=627)

Performing chain-level BOUNDARY path test on selected UUTs:

001c01 SIC 0x0 ... passed

001n02 SIC 0x0 ... passed

Performing system-level interconnect test on given configuration:

(16 of 16 patterns will be applied)

+++++

No errors detected during interconnect test.

Normal Program Termination

4.4 scan_kcar_l1 Script

The `scan_kcar_l1` script runs `scantest` on a G brick in an SGI 3000 series system. The G brick must contain two geometry engine (GE) boards (GE0 and GE1) and one Ktown2 board. `scan_kcar_l1` uses the following syntax:

```
scan_kcar_l1
```

Perform the following procedure to run this script:

1. Power up the system.
2. Log into the L3 system controller as `sgidiag` (default password: `sgi!diag`).
3. Enter the `cd /stand/sysco/usr` command to change to the `/stand/sysco/usr` directory.
4. Connect to the L2 system controller.

- To use a hardware L2 connection:

Connect the L3 system controller to the L2 system controller with an Ethernet connection via a crossover cable.

Obtain the IP address of the L2 system controller:

```
cat /var/state/dhcp/dhcpd.leases
```

Enter the command: `/stand/sysco/l2term --l2 <host>`

Example: `/stand/sysco/l2term --l2 10.1.1.37`

- To use the L2 emulator:

Enter the command: `/stand/sysco/bin/l2`

5. Configure the G brick to test as rack 2, slot 1.
6. Enter `/stand/sysco/bin/scan_kcar_l1` to start the script.

The script automatically runs the following tests:

- Boundary scan interconnect test on all boards
- Boundary scan interconnect test on the GE0 board
- Boundary scan interconnect test on the GE1 board
- Boundary scan interconnect test on the Ktown2 board

7. Interpret the output. The script returns output to the screen and to the following files in the directory from which you started the script:

- `g001.dmp` (`scantest` output from testing all boards)
- `g001_ge0.dmp` (`scantest` output from testing the GE0 board)
- `g001_ge1.dmp` (`scantest` output from testing the GE1 board)
- `g001_kt.dmp` (`scantest` output from testing the Ktown2 board)

Example:

```
/stand/sysco/usr>/stand/sysco/bin/scan_kcar_l1
```

Script to verify scan functionality of SN1 KCAR_L1

The L2 software must be running before starting tests.
The G-brick must be configured with RACK=002 and SLOT=01.
The G-brick must contain GE0, GE1 and KTOWN2 boards.

```
----- TEST WITH ALL BOARDS (GE0, GE1, KTOWN2) ENABLED -----  
Press ENTER to start scan integrity tests...  
<Enter>
```

Boundary-Scan Test Software (scantest V3.13) Tue Jan 15 10:48:39 2002

Loading chain configuration and reference data
File: /stand/sysco/cfg/scan/g001.cfg

Loading reference data for each board definition
Path: /stand/sysco/data/scan/cmp

Initializing JTAG communications to scan controller(s)
No hardware; running in software simulation/emulation mode

Initializing chain TAPs and configuring SICs:
002g01 with SIC at address 0x00 (CER=0x07)

Testing Integrity of Boundary Scan Instruction Registers
Performing IR length test on active UUTs:
002g01 SIC 0x0 ... passed (chain length=16)

Performing IR path test on active UUTs:
002g01 SIC 0x0 ... passed

Testing Integrity of Boundary Scan Bypass Registers
Performing BYPASS length test on selected UUTs:
002g01 SIC 0x0 ... passed (chain length=4)

Performing BYPASS path test on selected UUTs:
002g01 SIC 0x0 ... passed

Testing Integrity of Boundary Scan Data Registers
Performing chain-level BOUNDARY length test on selected UUTs:
002g01 SIC 0x0 ... passed (chain length=784)

Performing chain-level BOUNDARY path test on selected UUTs:
002g01 SIC 0x0 ... passed

Normal Program Termination
Results stored in output file: g001.dmp

```
----- TEST WITH GE0 BOARD ENABLED -----
Press ENTER to start scan integrity tests...
<Enter>

Boundary-Scan Test Software (scantest V3.13) Tue Jan 15 10:48:41 2002

Loading chain configuration and reference data
  File: /stand/sysco/cfg/scan/g001_ge0.cfg

Loading reference data for each board definition
  Path: /stand/sysco/data/scan/cmp

Initializing JTAG communications to scan controller(s)
  No hardware; running in software simulation/emulation mode

Initializing chain TAPs and configuring SICs:
002g01 with SIC at address 0x00 (CER=0x04)

Testing Integrity of Boundary Scan Instruction Registers
Performing IR length test on active UUTs:
002g01 SIC 0x0 ... passed (chain length=5)

Performing IR path test on active UUTs:
002g01 SIC 0x0 ... passed

Testing Integrity of Boundary Scan Bypass Registers
Performing BYPASS length test on selected UUTs:
002g01 SIC 0x0 ... passed (chain length=1)

Performing BYPASS path test on selected UUTs:
002g01 SIC 0x0 ... passed

Testing Integrity of Boundary Scan Data Registers
Performing chain-level BOUNDARY length test on selected UUTs:
002g01 SIC 0x0 ... passed (chain length=264)

Performing chain-level BOUNDARY path test on selected UUTs:
002g01 SIC 0x0 ... passed

Normal Program Termination
Results stored in output file: g001_ge0.dmp
```

```
----- TEST WITH GE1 BOARD ENABLED -----
Press ENTER to start scan integrity tests...
<Enter>

Boundary-Scan Test Software (scantest V3.13) Tue Jan 15 10:48:42 2002

Loading chain configuration and reference data
  File: /stand/sysco/cfg/scan/g001_ge1.cfg

Loading reference data for each board definition
  Path: /stand/sysco/data/scan/cmp

Initializing JTAG communications to scan controller(s)
  No hardware; running in software simulation/emulation mode
```

```
Initializing chain TAPs and configuring SICs:
002g01 with SIC at address 0x00 (CER=0x01)

Testing Integrity of Boundary Scan Instruction Registers
Performing IR length test on active UUTs:
002g01 SIC 0x0 ... passed (chain length=5)

Performing IR path test on active UUTs:
002g01 SIC 0x0 ... passed

Testing Integrity of Boundary Scan Bypass Registers
Performing BYPASS length test on selected UUTs:
002g01 SIC 0x0 ... passed (chain length=1)

Performing BYPASS path test on selected UUTs:
002g01 SIC 0x0 ... passed

Testing Integrity of Boundary Scan Data Registers
Performing chain-level BOUNDARY length test on selected UUTs:
002g01 SIC 0x0 ... passed (chain length=264)

Performing chain-level BOUNDARY path test on selected UUTs:
002g01 SIC 0x0 ... passed

Normal Program Termination
Results stored in output file: g001_ge1.dmp

----- TEST WITH KTOWN2 BOARD ENABLED -----
Press ENTER to start scan integrity tests...
<Enter>

Boundary-Scan Test Software (scantest V3.13) Tue Jan 15 10:48:43 2002

Loading chain configuration and reference data
  File: /stand/sysco/cfg/scan/g001_kt.cfg

Loading reference data for each board definition
  Path: /stand/sysco/data/scan/cmp

Initializing JTAG communications to scan controller(s)
  No hardware; running in software simulation/emulation mode

Initializing chain TAPs and configuring SICs:
002g01 with SIC at address 0x00 (CER=0x02)

Testing Integrity of Boundary Scan Instruction Registers
Performing IR length test on active UUTs:
002g01 SIC 0x0 ... passed (chain length=6)

Performing IR path test on active UUTs:
002g01 SIC 0x0 ... passed

Testing Integrity of Boundary Scan Bypass Registers
Performing BYPASS length test on selected UUTs:
002g01 SIC 0x0 ... passed (chain length=2)
```

Performing BYPASS path test on selected UUTs:

002g01 SIC 0x0 ... passed

Testing Integrity of Boundary Scan Data Registers

Performing chain-level BOUNDARY length test on selected UUTs:

002g01 SIC 0x0 ... passed (chain length=256)

Performing chain-level BOUNDARY path test on selected UUTs:

002g01 SIC 0x0 ... passed

Normal Program Termination

Results stored in output file: g001_kt.dmp

scantool Application

This chapter describes the `scantool` application. `scantool` enables you to control scan operations from Tcl/Tk scripts. These scripts can control the following test functions that are accessible through the JTAG/scan hardware:

- Memory built-in self-test (BIST)
- Clock control
- Auxiliary access to memory-mapped registers (MMRs)
- Internal latch dumping
- Debug port control

`scantool` includes graphical user interface and command-line modes.

5.1 Available Scripts

Table 5-1 through Table 5-4 describe useful scripts that are available in the current diagnostic release.

Table 5-1 IP35 C-brick Scripts (Located in `/stand/sysco/data/scan/diags/ip35`)

Script	Description
<code>ip35_debug.tcl</code>	Configures the debug port MUX to steer a selected set of 32 signals to the debug port of the Bedrock ASIC where the signals can be viewed by a logic analyzer (This script is not used in the field.)
<code>ip35_dump_br_latches.tk</code>	Dumps the Bedrock ASIC latches from an IP35 C brick (This script could be used with the FRU analyzer to dump the state of the Bedrock ASIC when a brick does not respond to an NML.)
<code>ip35_llp.tcl</code>	Configures and runs a simple LLP test packet by using the at-speed LLP test feature (This script is not used in the field.)
<code>ip35_membist.tcl</code>	Runs the on-board memory built-in self test (BIST) feature on the Bedrock ASIC (This script is not used in the field.)

Table 5-2 R-brick Scripts (Located in /stand/sysco/data/scan/diags/rbrick)

Script	Description
r_debug.tcl	Configures the debug port MUX to steer a selected set of 32 signals to the debug port where the signals can be viewed by a logic analyzer (This script is not used in the field.)
r_dump_latches.tcl	Dumps the router ASIC latches
r_stopclk.tcl	Stops the router clock

Table 5-3 SGI Origin 300 Base Module Scripts (Located in /stand/sysco/data/scan/diags/ip45)

Script	Description
ip45_dump_br_latches.tk	Dumps the Bedrock ASIC latches from a SGI Origin 300 base module (This script could be used with the FRU analyzer to dump the state of the Bedrock ASIC when a brick does not respond to an NMI.)
ip45_membist.tcl	Runs the on-board memory built-in self test (BIST) feature on the Bedrock ASIC (This script is not used in the field.)

Table 5-4 PCI Expansion Module Scripts (Located in /stand/sysco/data/scan/diags/pebrick)

Script	Description
pe_dump_br_latches.tk	Dumps the Bedrock ASIC latches from a SGI Origin 300 PCI expansion module (This script could be used with the FRU analyzer to dump the state of the Bedrock ASIC when a brick does not respond to an NMI.)

5.2 scantool Command-line Options

scantool uses the following command syntax:

```
scantool [-h] [-l] [-c <config_file>] [-f <script_name>] [-g]
[-n | -s | -0 | -1] [--l2 <host> | -- interface <iface> | --ssn <serial>[:<rack>]
| --sysname <system_name>[:<rack>] | --serial <host>:<port> |
--dev <device_name>]
```

Table 5-5 describes the command-line options that you can use to control scantool.

Table 5-5 scantool Command-line Options

Option	Description
-h	Displays help information
-l	Starts scantool in command-line mode (a prompt appears at which you can enter scantool commands) This option is required for command-line mode.
-c <config_file>	Selects the configuration file to use This option is required for command-line mode.
-f <script_name>	Selects the script to run You can also use the graphical user interface to select which script to run.
-g	Enables scantest to run graphical Tk scripts in command-line mode
-n	Sets the communication method to “null” mode: scantool runs the script but does not send the commands to the scan hardware. No scan data is received from the hardware.
-s	Sets the communication method to “simulation” mode: scantool connects to a Verilog simulator through a socket connection. (This configuration is not used in the field.)
-0	Sets the communication method to “SN0” mode: scantool connects to an external system through the parallel port of a Silicon Graphics Indy visual workstation. (This configuration is not used in the field.)
-1	Sets the communication method to “SN1” mode: scantool connects to an SGI 3000 series system through an L3 system controller. The L3 system controller must be connected to an L2 system controller or it must be running the L2 emulator. This is the default communication method.
--l2 <host>	Connects to the L2 system controller with the specified host name or IP address

Table 5-5 (continued) scantool Command-line Options

Option	Description
<code>--interface <iface></code>	Connects to the L2 system controller with the specified network interface (for example, eth0)
<code>--ssn <serial>[:<rack>]</code>	Connects to the system with the specified serial number (for example, L01234567)
<code>--sysname <system_name>[:<rack>]</code>	Connects to the system specified by <system_name>
<code>--serial <host>:<port></code>	Connects to the specified Ethernet host name and port
<code>--dev <device_name></code>	Connects to the specified local serial port device

5.3 Running Scripts from a Graphical User Interface

Perform the following procedure to run `scantool` scripts from the graphical user interface on a single brick:

1. If the system is not running, power up the system.
2. Log into the L3 system controller as `sgidiag` (default password: `sgi!diag`).
3. Enter the `cd /stand/sysco/usr` command to change to the `/stand/sysco/usr` directory.
4. Connect to the L2 system controller.
 - To use a hardware L2 connection:

Connect the L3 system controller to the L2 system controller with an Ethernet connection by using a crossover cable.

Obtain the IP address of the L2 system controller:

```
cat /var/state/dhcp/dhcpd.leases
```

Enter the command: `/stand/sysco/l2term --l2 <host>`
 - To use the L2 emulator:

Enter the command: `/stand/sysco/bin/l2`
5. Enter `scantool [test_options]` at the L3 system controller prompt.

Refer again to Table 5-5 for descriptions of the available command-line options.

Figure 5-1 shows the `scantool` graphical user interface (also called the graphical console).



Figure 5-1 scantool Graphical User Interface (Initial Window)

6. Choose **File** -> **Set SINGLE brick Config.** (The interface displays the **Configure One Brick** window; refer to Figure 5-2.)

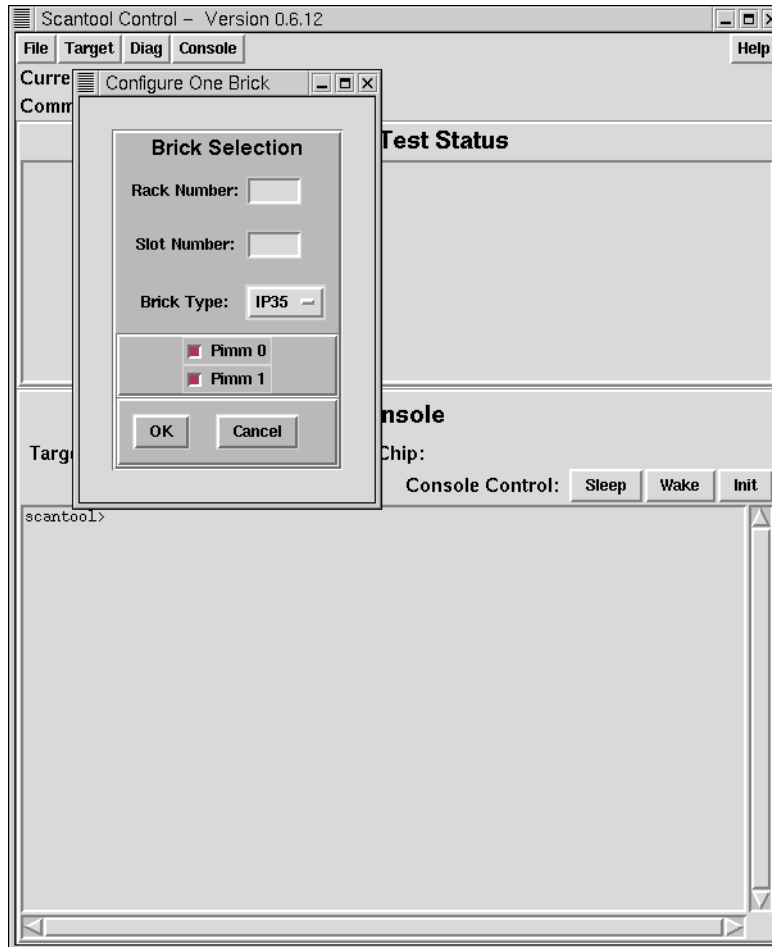


Figure 5-2 Selecting the Brick to Test

7. Select the brick that you want to test (refer again to Figure 5-2):

- Enter the rack number in the **Rack Number** field.
- Enter the slot number in the **Slot Number** field.

If you do not know the rack and slot locations of the brick that you want to test, use the `L2 cfg` command to determine the bricks that are available.

- Select the brick type from the **Brick Type** menu.

If you are testing an IP35 C brick, you can select the PIMMs to test (none, PIMM0, and/or PIMM1).

Note: The IP37 selection is not valid for field use.

Figure 5-3 shows example settings used to select PIMMs 0 and 1 in an IP35 C brick located in slot 11 of rack 7.

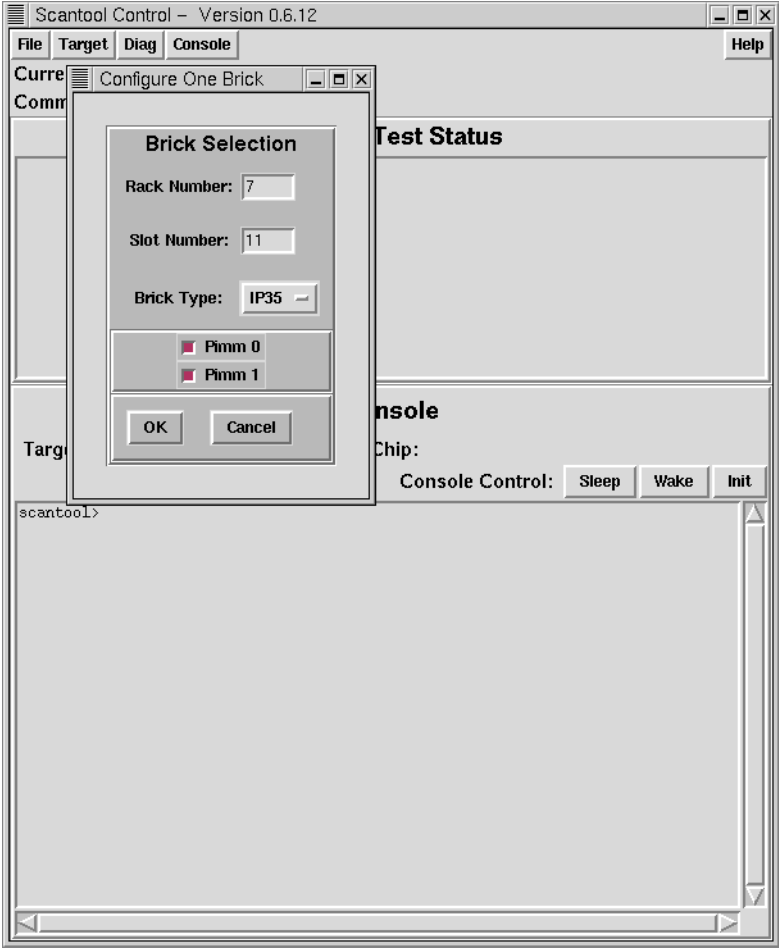


Figure 5-3 Selecting the Brick (Example Settings)

8. Click on the OK button.

Figure 5-4 shows the interface with the brick selected. Notice that the Targeted Module field shows the brick from this example (007c11 = C brick in slot 11 of rack 7).



Figure 5-4 Interface with Brick Selected

9. Choose **Diag** -> **Start Console Diag** to load a script.

Note: If you do not want to run a script, you can enter individual `scantool`, `Tcl`, and `Tk` commands at the `scantool>` prompt.

The interface displays the `Open` window. (Refer to Figure 5-5.) Use this window to select the script to run.

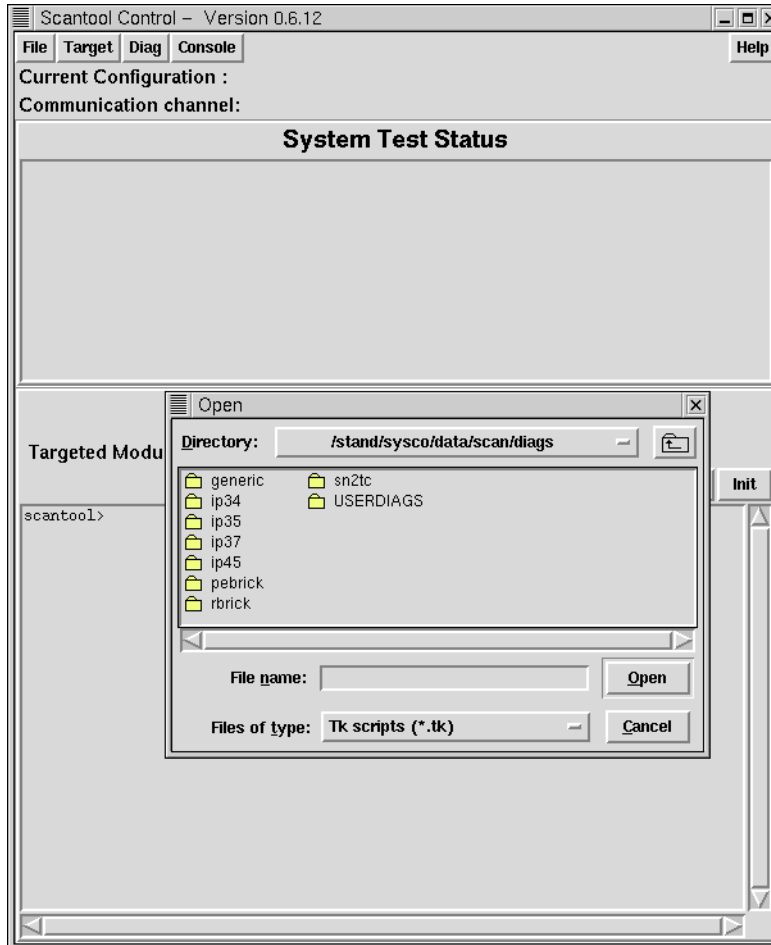


Figure 5-5 Selecting the Script to Run

10. Select the script that you want to run.

- To select a Tcl script, set the Files of type field to Tcl scripts. Then, select the script from the appropriate directory. (Refer Figure 5-6.)
- To select a Tk script, set the Files of type field to Tk scripts. Then, select the script from the appropriate directory. (Refer to Figure 5-7.)

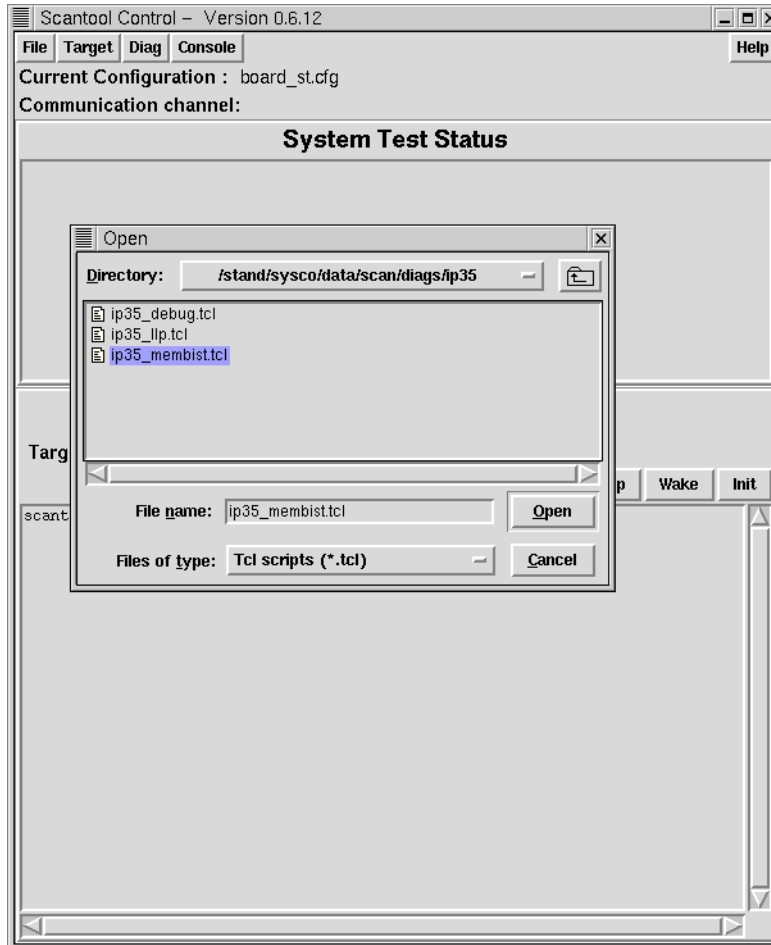


Figure 5-6 Selecting a Tcl Script

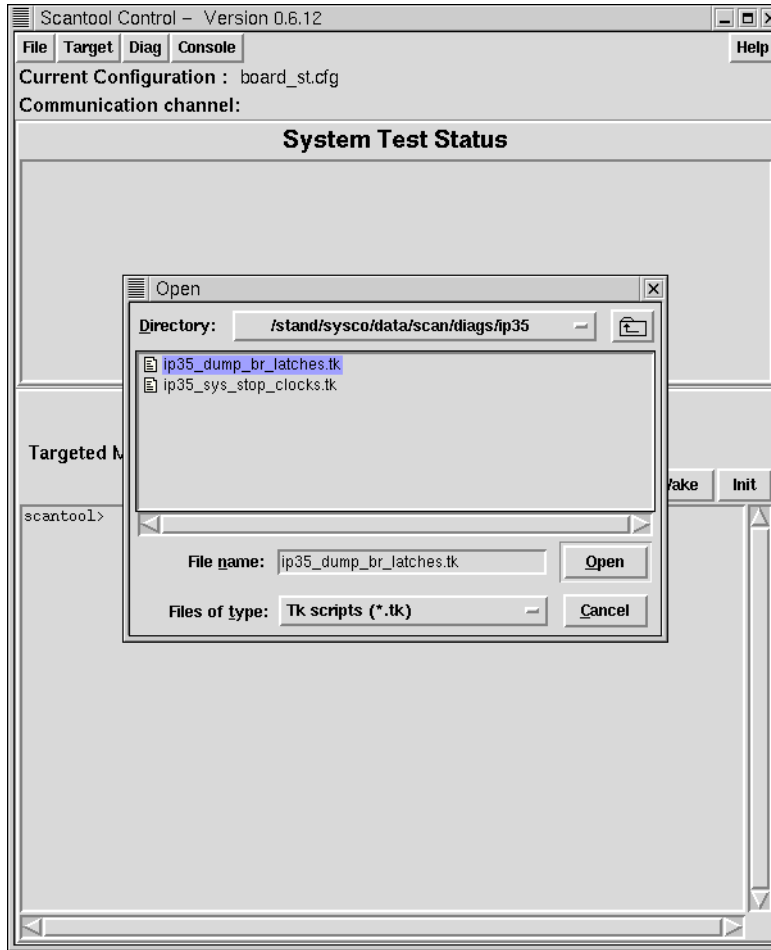


Figure 5-7 Selecting a Tk Script

If you select a Tcl script, the output from the script appears in the Console portion of the window. (Refer to Figure 5-8.)

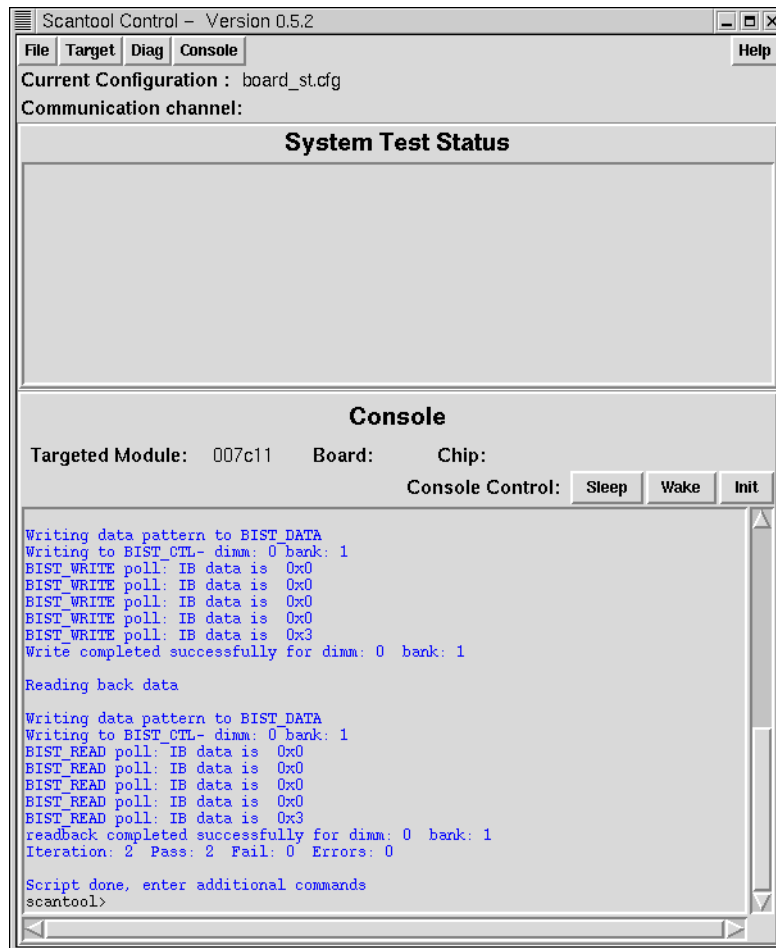


Figure 5-8 Running a Tcl Script

If you select a Tk script, the script displays a new window that you can use to set parameters. (Refer to Figure 5-9.) Set the parameters, and click GO to run the script. Output from the script appears in the window. (Refer to Figure 5-10.)

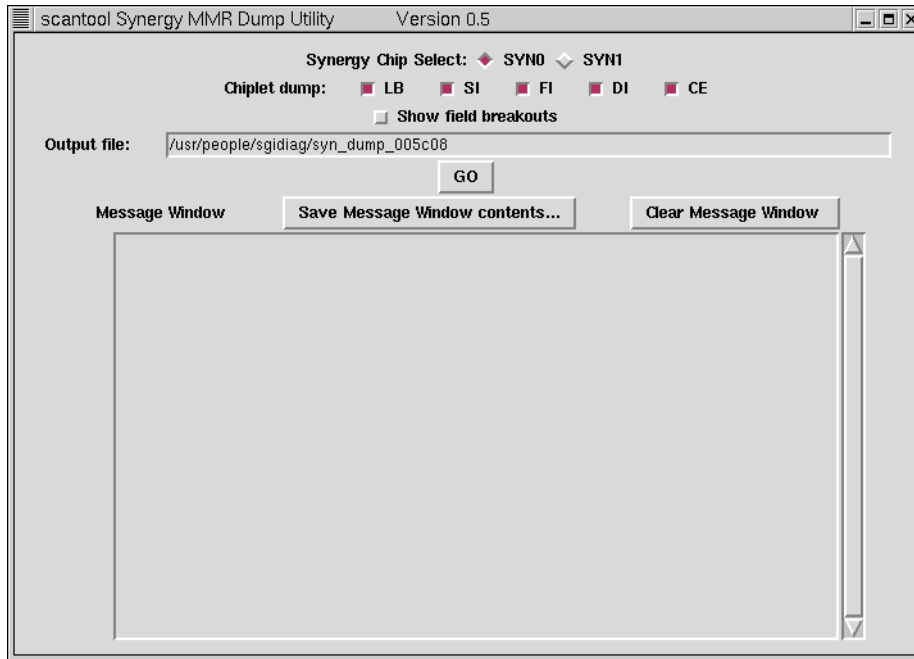


Figure 5-9 Running a Tk Script (Example Window)

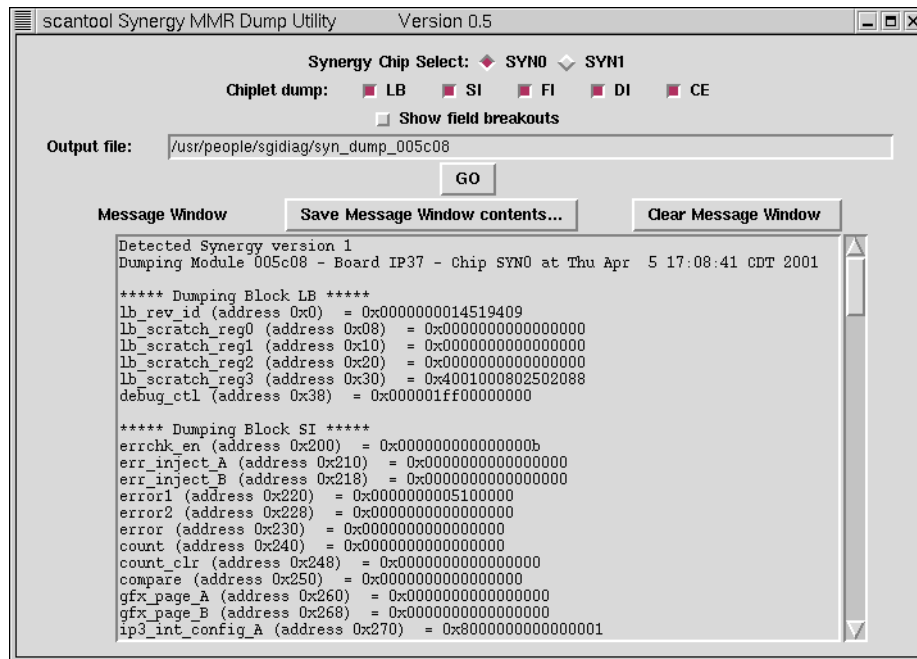


Figure 5-10 Running a Tk Script (Example Output)

- When you finish running scripts, close any windows that `scantool` opened and then choose `File -> Exit`.

Note: Normally, you do not need to reset the system after using `scantool`. However, if you dump the internal latches (typically after a system hang), you should enter the `L2 reset` command to reset the system.

5.4 Running Scripts from the Command Line

Perform the following procedure to run `scantool` scripts from the command line:

- If the system is not running, power up the system.
- Log into the L3 system controller as `sgidiag` (default password: `sgi!diag`).
- Enter the `cd /stand/sysco/usr` command to change to the `/stand/sysco/usr` directory.
- Connect to the L2 system controller.
 - To use a hardware L2 connection:

Connect the L3 system controller to the L2 system controller with an Ethernet connection by using a crossover cable.

Obtain the IP address of the L2 system controller:

```
cat /var/state/dhcp/dhcpd.leases
```

Enter the command: `/stand/sysco/l2term --l2 <host>`
 - To use the L2 emulator:

Enter the command: `/stand/sysco/bin/l2`
- Ensure that the required environment variables are properly set. (Refer to Table 5-6.)

Table 5-6 scantool Environment Variables

Variable	Setting
<code>SCAN_PATH_BIN</code>	<code>/stand/sysco/bin</code>
<code>SCAN_PATH_CONFIG</code>	<code>/stand/sysco/cfg</code>
<code>SCAN_PATH_DATA</code>	<code>/stand/sysco/data/scan</code>
<code>SCAN_USER_BIN</code>	<code>/stand/sysco/usr/bin</code>
<code>SCAN_USER_CONFIG</code>	<code>/stand/sysco/usr/cfg</code>
<code>SCAN_USER_DATA</code>	<code>/stand/sysco/usr/data</code>
<code>SCAN_USER_DIAGS</code>	<code>/stand/sysco/usr/data/diags</code>
<code>LD_LIBRARY_PATH</code>	<code>/stand/sysco/lib</code>
<code>TCL_LIBRARY</code>	<code>/stand/sysco/lib</code>

6. Select the configuration file to use. (Refer to Table 5-7.)

Table 5-7 scantool Configuration Files

File	Description
001c10_st.cfg	Targets the C brick in slot 10 of rack 1
001c13_st.cfg	Targets the C brick in slot 13 of rack 1
001c16_st.cfg	Targets the C brick in slot 16 of rack 1
001c21_st.cfg	Targets the C brick in slot 21 of rack 1
001c24_st.cfg	Targets the C brick in slot 24 of rack 1
001c29_st.cfg	Targets the C brick in slot 29 of rack 1
001c32_st.cfg	Targets the C brick in slot 32 of rack 1
001c35_st.cfg	Targets the C brick in slot 35 of rack 1
001r19_st.cfg	Targets the R brick in slot 19 of rack 1
001r27_st.cfg	Targets the R brick in slot 27 of rack 1
ip35sys_8p_st.cfg	Targets bricks in default configuration for an 8-processor system
ip35sys_16p_st.cfg	Targets bricks in default configuration for a 16-processor system
ip35sys_32p_st.cfg	Targets bricks in default configuration for a 32-processor system
ip35sys_64p_st.cfg	Targets bricks in default configuration for a 64-processor system
ip35sys_128p_st.cfg	Targets bricks in default configuration for a 128-processor system
ip35sys_256p_st.cfg	Targets bricks in default configuration for a 256-processor system
ip45_002_st.cfg	Targets components in an SGI Origin 300 system

Note: If you need to modify a configuration file, copy the file to `$$SCAN_USER_CONFIG/user_st.cfg` and modify it. Then, use the `-c` command-line option to load the modified file.

7. Select (or create) the script that you want to run. (Refer again to Table 5-1 through Table 5-4 for the scripts that are available in the diagnostic release.)
8. Enter the `scantool -l -c <file> [test_options]` command at the L3 system controller prompt.

For example, the following command line starts `scantool` and runs the `ip35_membist.tcl` script on the IP35 C brick in slot 16 of rack 1 of an SGI 3000 system:

```
pyro> scantool -l -c $$SCAN_PATH_CONFIG/scan/001c16_st.cfg -f  
$$SCAN_PATH_DATA/diags/ip35/ip35_membist.tcl
```

Refer again to Table 5-5 for descriptions of the available command-line options.

9. Interpret the output.

Note: Normally, you do not need to reset the system after using `scantool`. However, if you dump the internal latches (typically after a system hang), you should enter the `L2 reset` command to reset the system.

5.5 scantool Script Commands

You can copy the standard `scantool` scripts and customize them for your site. You can also create new `scantool` scripts. (Table 5-8 shows the script commands that are available.)

Note: To modify a script, you must copy it to a user-writable directory (for example, `/stand/sysco/usr` or `/usr/people/sgidiag`).

Table 5-8 scantool Script Commands

Command	Description
SEL	Selects an object
DSEL	Deselects an object
INST	Sets the current instruction
GOINST	Scans the current instruction into a scan chain
DATA	Sets the current data pattern or reads data
GODATA	Scans the current data pattern into a scan chain
REG	Specifies a user-defined data pattern
REGDATA	Sets the user-defined data
RESET	Resets the TAP controller(s) on the current scan chain
SIC	Causes a “family” to access and manipulate the scan interface chip (SIC)
GORUN	Moves the TAP controller to the run-test-idle state and toggles the TCK signal
GOIDLE	Moves the TAP controller to the run-test-idle state
INFO	Causes a “family” to extract information from an entity
SIMACC	Causes a “family” to control a Verilog simulation
EXIT	Exits the script

Refer to the following URL for more information about these commands:

http://wwwcf.americas.sgi.com/~kermes/scantool/scantool_toc.html

5.6 Dumping the Internal Bedrock Latches

If a system hangs and does not respond to an NMI, it is sometimes useful to dump the internal Bedrock latches. You may be requested to collect this data and return it to SGI for further analysis.

There are `scantool` scripts available to dump the internal Bedrock latches from the Bedrock ASICs in the following components:

- SGI 3000 series system C brick
- SGI Origin 300 base module
- SGI Origin 300 PCI expansion module

5.6.1 SGI 3000 Series System C Brick

To dump the internal Bedrock latches from the Bedrock ASIC in an SGI 3000 series C brick, perform the following procedure:

1. Connect a USB cable from the L3 system controller to the SGI 3000 series system.
2. Log into the L3 system controller as `sgidiag` (default password: `sgi!diag`).
3. Enter the `cd /stand/sysco/usr` command to change to the `/stand/sysco/usr` directory.
4. Connect to the L2 system controller.

- To use a hardware L2 connection:

Connect the L3 system controller to the L2 system controller with an Ethernet connection by using a crossover cable.

Obtain the IP address of the L2 system controller:

```
cat /var/state/dhcp/dhcpd.leases
```

Enter the command: `/stand/sysco/l2term --l2 <host>`

Example: `/stand/sysco/l2term --l2 10.1.1.37`

- To use the L2 emulator:

Enter the command: `/stand/sysco/bin/l2`

5. Start the `scantool` application:

```
/stand/sysco/bin/scantool --sysname <system_name>
```

Example: `/stand/sysco/bin/scantool --sysname lefty`

Figure 5-11 shows the `scantool` graphical user interface (also called the graphical console).

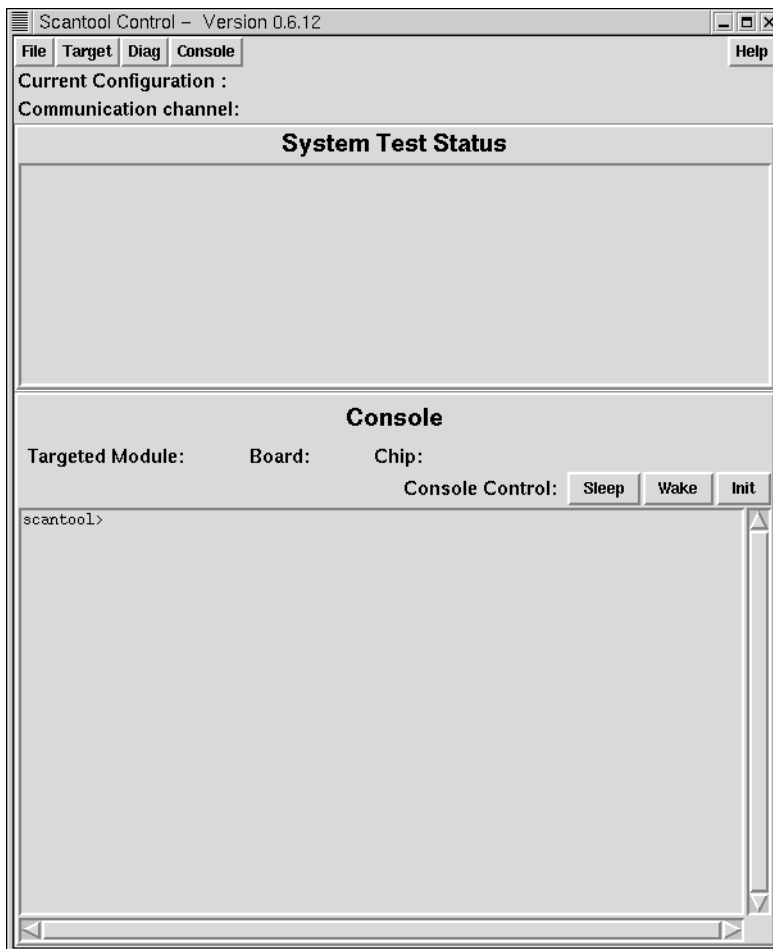


Figure 5-11 scantool Graphical User Interface (Initial Window)

6. Choose **File** -> **Set SINGLE brick Config.**

The interface displays the **Configure One Brick** window. (Refer to Figure 5-12.)

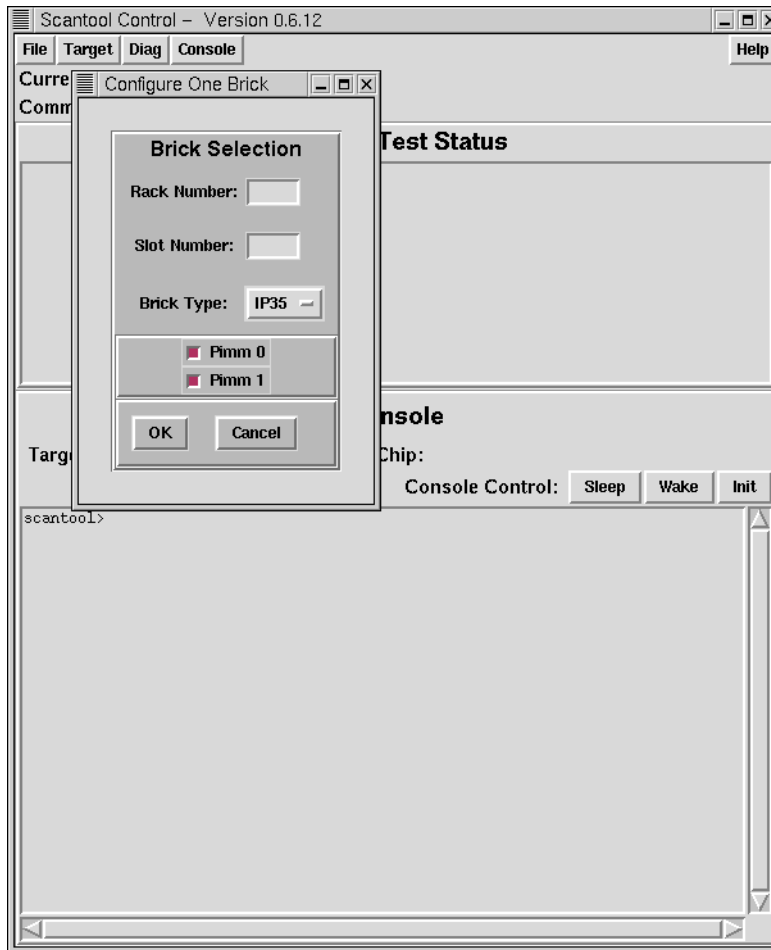


Figure 5-12 Configure One Brick Window

7. Specify the IP35 C brick that you want to use (refer again to Figure 5-12):

- Enter the rack number in the Rack Number field.
- Enter the slot number in the Slot Number field.

If you do not know the rack and slot locations of the brick that you want to test, use the `L2 cfg` command to determine the bricks that are available.

- Select IP35 from the Brick Type menu.

Figure 5-13 shows example settings used to select the C brick located in slot 11 of rack 7.

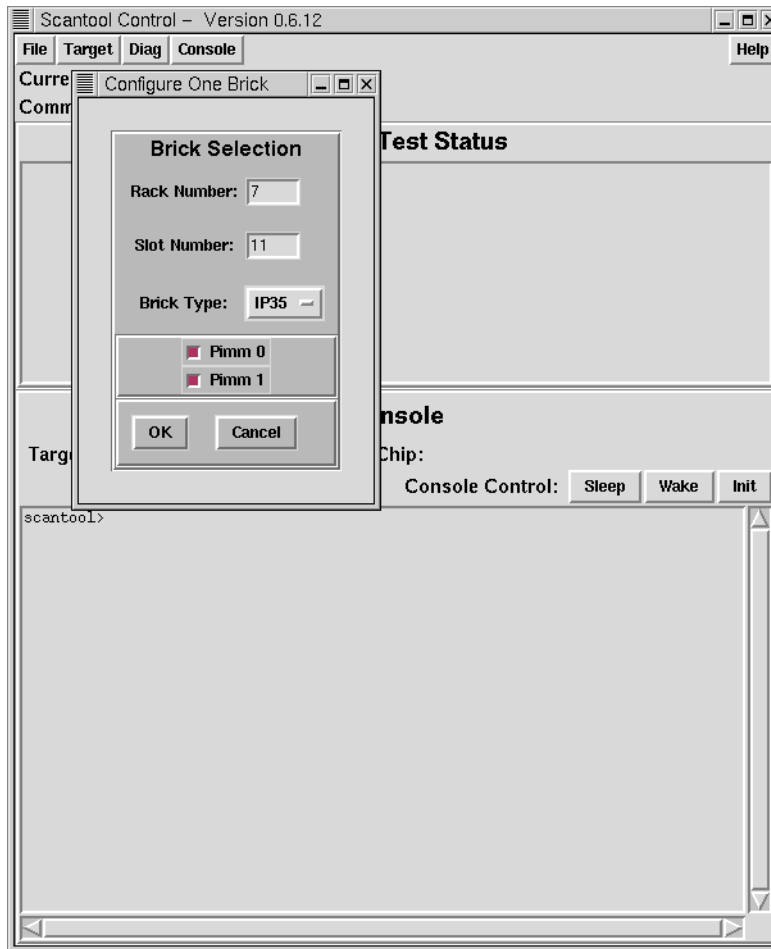


Figure 5-13 Selecting the IP35 C Brick to Use

8. Click on the OK button.
9. Choose **Diag** -> **Start Console Diag**.

The interface displays the `Open` window. (Refer to Figure 5-14.) Use this window to select the script to run.

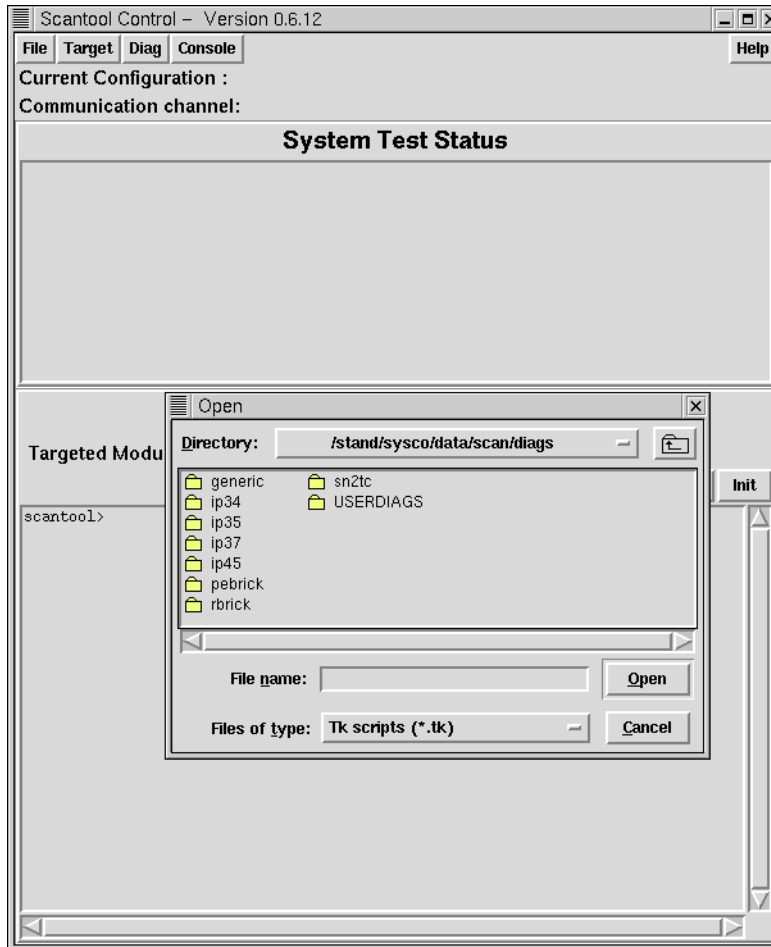


Figure 5-14 Selecting the IP35 C Brick Script to Run

10. Double-click on ip35.
11. Double-click on ip35_dump_br_latches.tk. (Refer to Figure 5-15.)

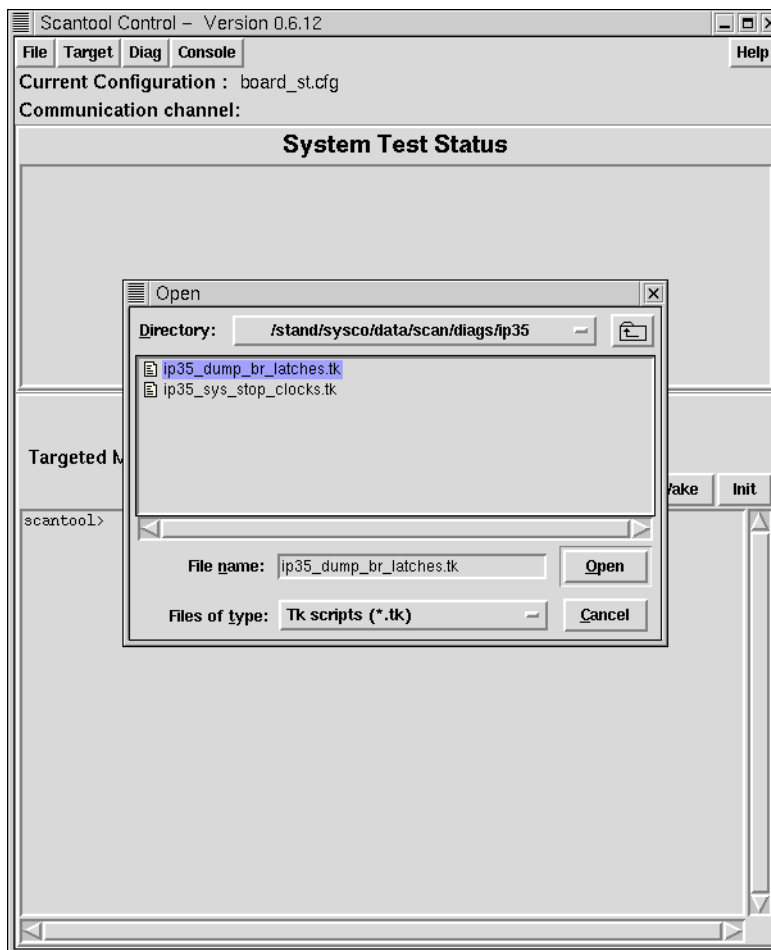


Figure 5-15 Selecting the ip35_dump_br_latches.tk Script

The interface displays the IP35 Bedrock Internal Latch Dump window. (Refer to Figure 5-16.)

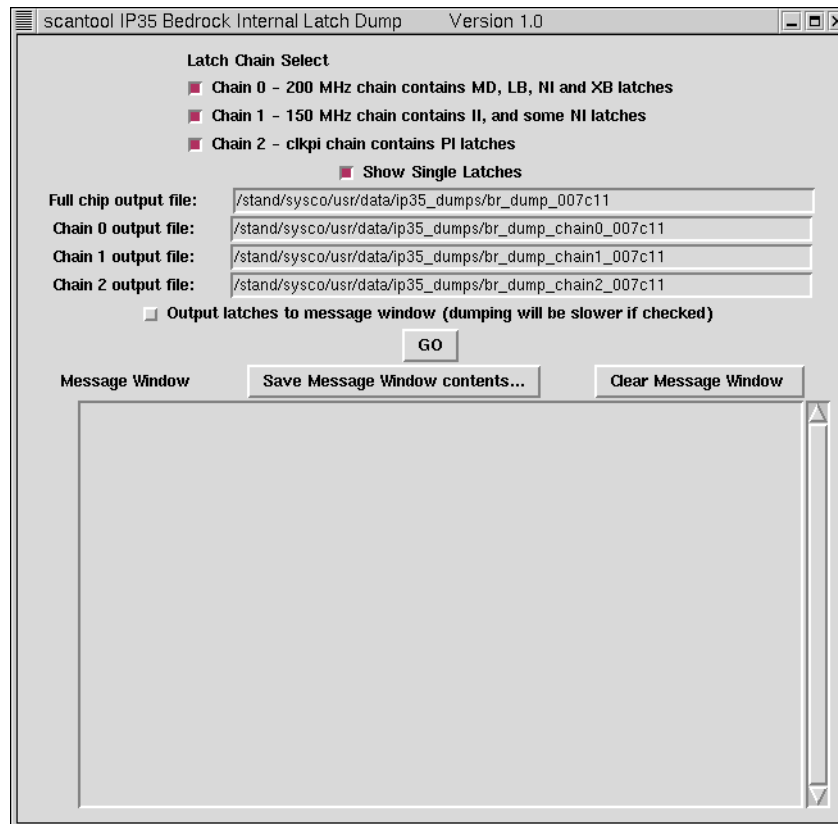


Figure 5-16 IP35 Bedrock Internal Latch Dump Window

12. Modify the parameters, if necessary.

Note: Normally, you can use the default parameter settings.

The parameters select the scan chains to use, the output files to use, and whether or not `scantool` should show single latches and output the latches to the Message Window.

13. Click on the GO button.

The interface displays a warning message. (Refer to Figure 5-17.)

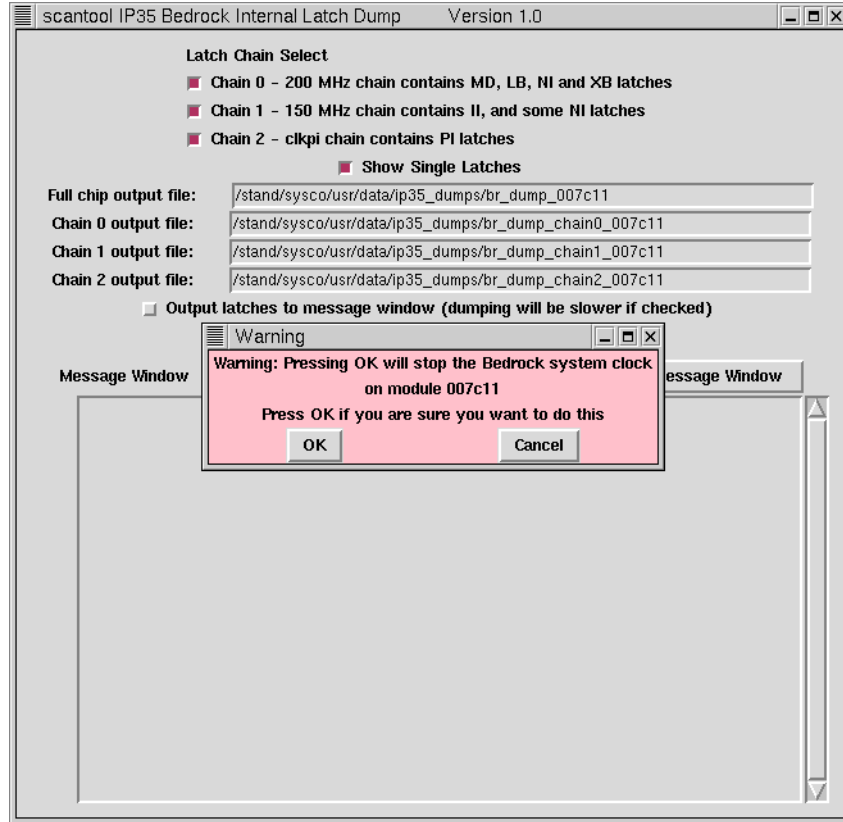


Figure 5-17 Bedrock System Clock Warning Message

14. Click on the OK button.

Output appears in the lower portion of the window. (Refer to Figure 5-18.) Output is also written to the specified files.

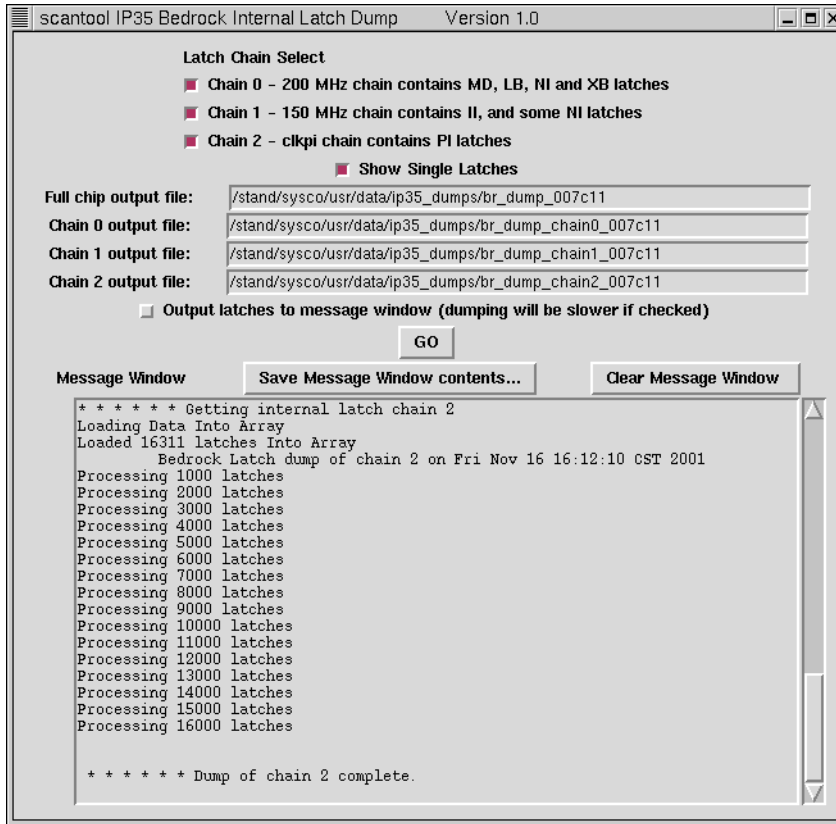


Figure 5-18 Output from the ip35_dump_latches.tk Script

15. Run the L1 reboot_11 command on the selected C brick.

16. Reset the system.

5.6.2 SGI Origin 300 Base Module

To dump the internal Bedrock latches from the Bedrock ASIC in an SGI Origin 300 base module, perform the following procedure:

1. Connect a USB cable from the L3 system controller to the SGI Origin 300 base module.
2. Log into the L3 system controller as `sgidiag` (default password: `sgi!diag`).
3. Enter the `cd /stand/sysco/usr` command to change to the `/stand/sysco/usr` directory.
4. Enter the following command to connect to the L2 emulator:

```
/stand/sysco/bin/l2
```

5. Enter the following command to start the `scantool` application:

```
/stand/sysco/bin/scantool --sysname <system_name>
```

Example: `/stand/sysco/bin/scantool --sysname lefty`

Figure 5-19 shows the `scantool` graphical user interface (also called the graphical console).

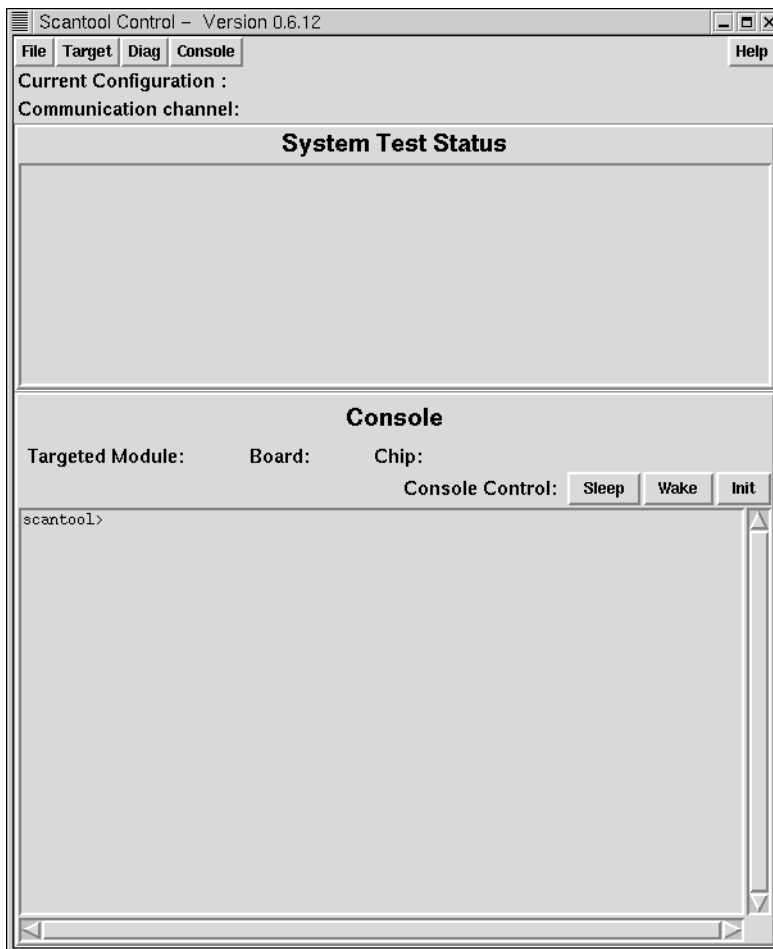


Figure 5-19 scantool Graphical User Interface (Initial Window)

6. Choose File -> Set SINGLE brick Config.

The interface displays the Configure One Brick window. (Refer to Figure 5-20.)

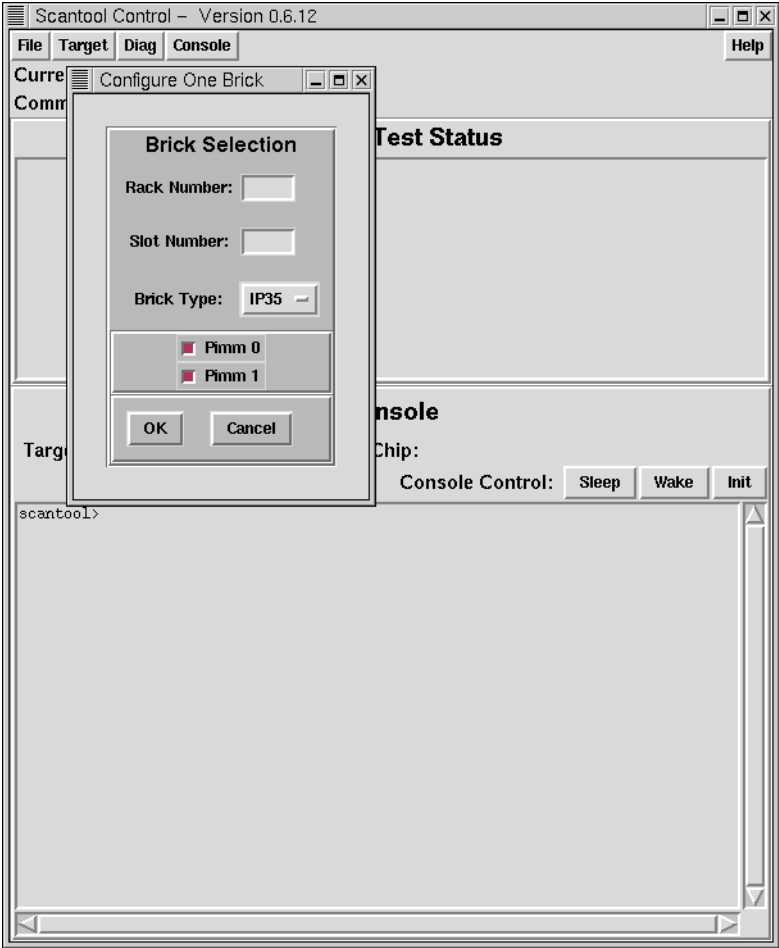


Figure 5-20 Configure One Brick Window

7. Specify the SGI Origin 300 base module that you want to use (refer again to Figure 5-20):

- Enter the rack number in the `Rack Number` field.
- Enter the slot number in the `Slot Number` field.

If you do not know the rack and slot locations of the brick that you want to test, use the `L2 cfg` command to determine the bricks that are available.

- Select the number CPUs.
- Select `IP45` from the `Brick Type` menu.

Figure 5-21 shows example settings used to select the SGI Origin 300 base module located in slot 19 of rack 1.

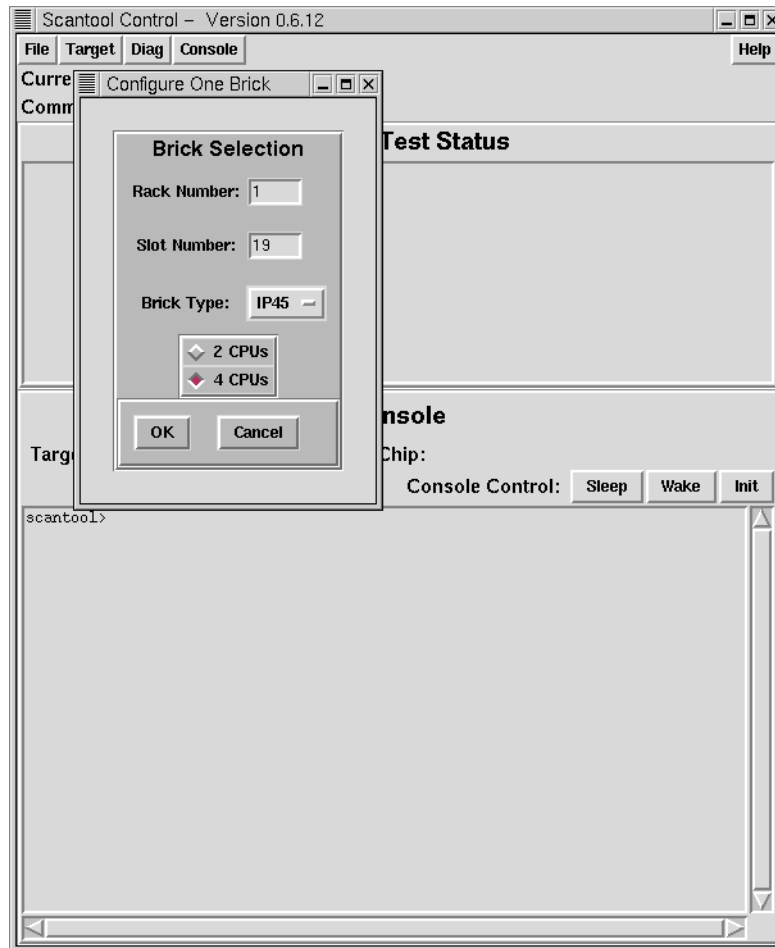


Figure 5-21 Selecting the SGI Origin 300 Base Module to Use

8. Click on the OK button.
9. Choose Diag -> Start Console Diag.

The interface displays the Open window. (Refer to Figure 5-22.) Use this window to select the script to run.

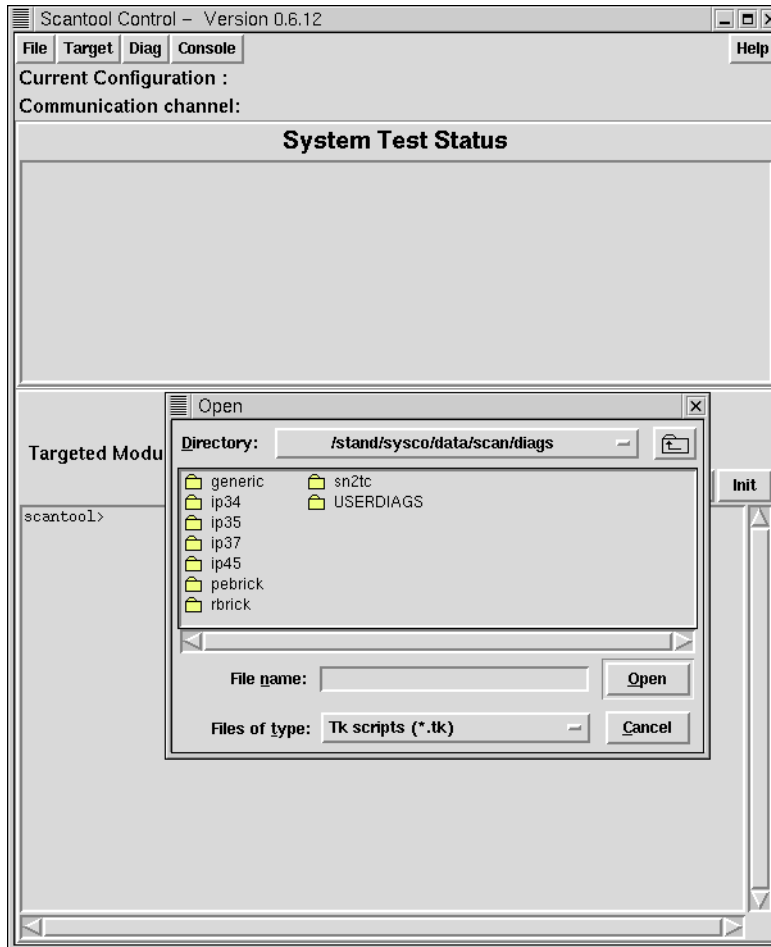


Figure 5-22 Selecting the SGI Origin 300 Base Module Script to Run

10. Double-click on ip45.
11. Double-click on ip45_dump_br_latches.tk. (Refer to Figure 5-23.)

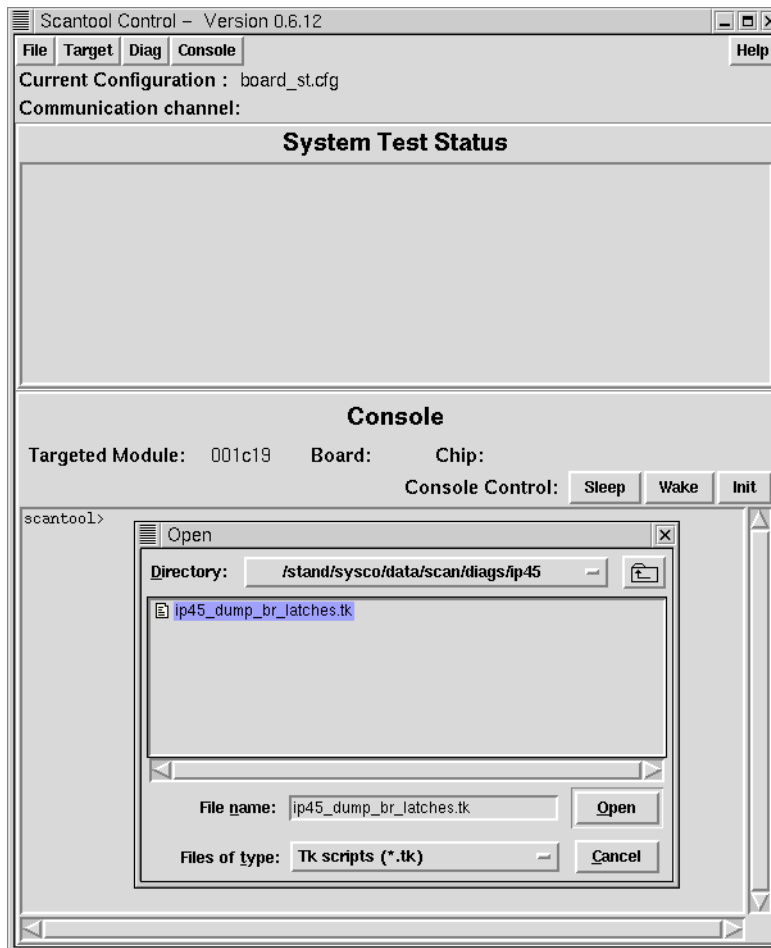


Figure 5-23 Selecting the ip45_dump_br_latches.tk Script

The interface displays the IP45 Bedrock Internal Latch Dump window. (Refer to Figure 5-24.)

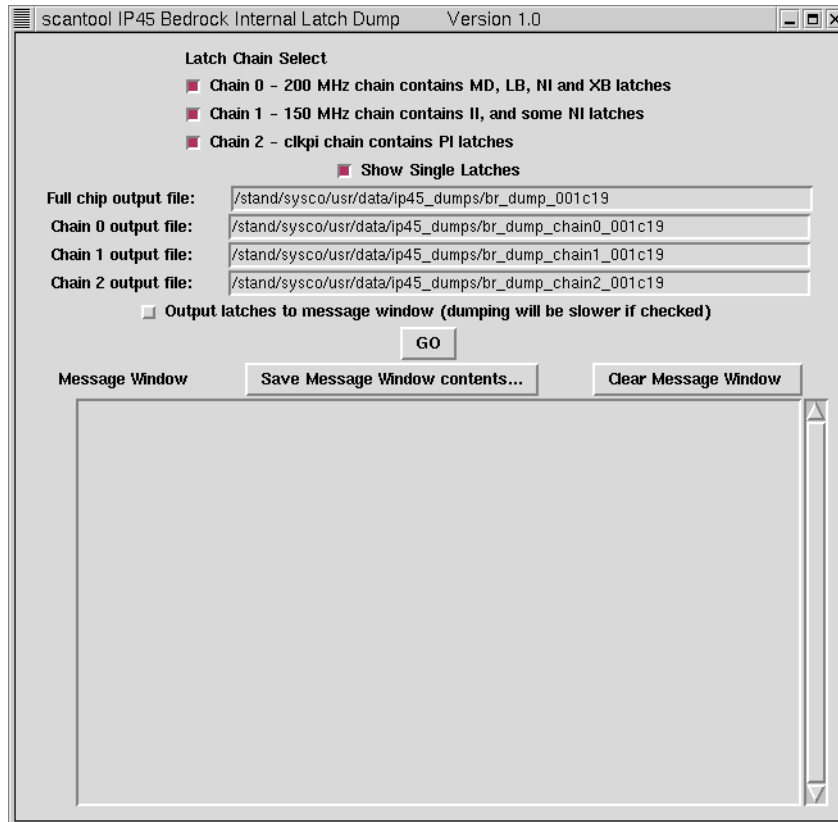


Figure 5-24 IP45 Bedrock Internal Latch Dump Window

12. Modify the parameters, if necessary.

Note: Normally, you can use the default parameter settings.

The parameters select the scan chains to use, the output files to use, and whether or not `scantool` should show single latches and output the latches to the Message Window.

13. Click on the GO button.

The interface displays a warning message. (Refer to Figure 5-25.)

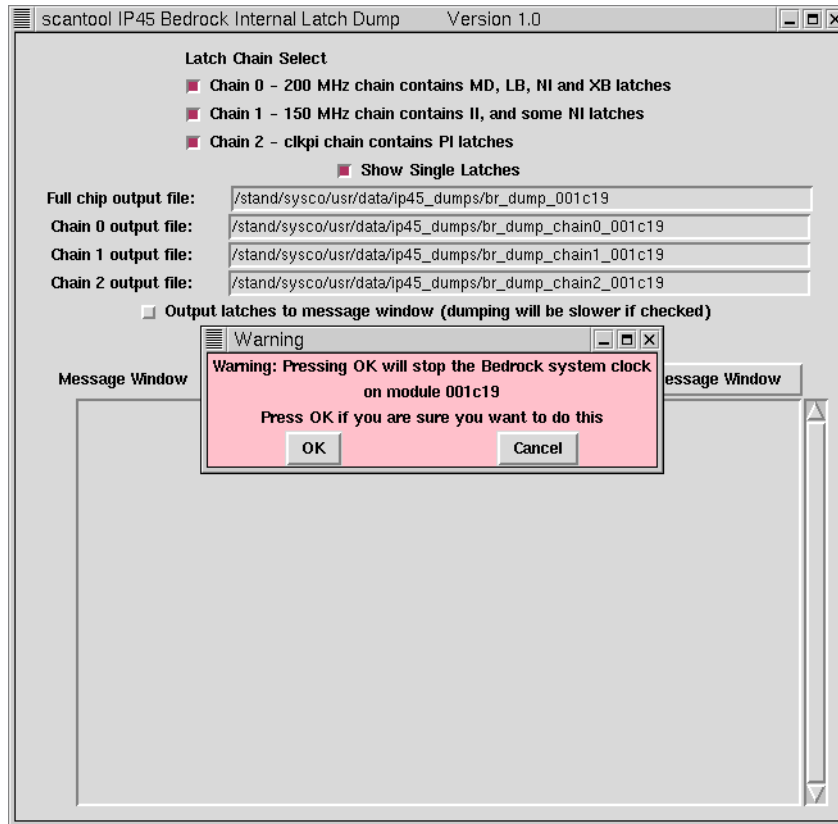


Figure 5-25 Bedrock System Clock Warning Message

14. Click on the OK button.

Output appears in the lower portion of the window. (Refer to Figure 5-26.) Output is also written to the specified files.

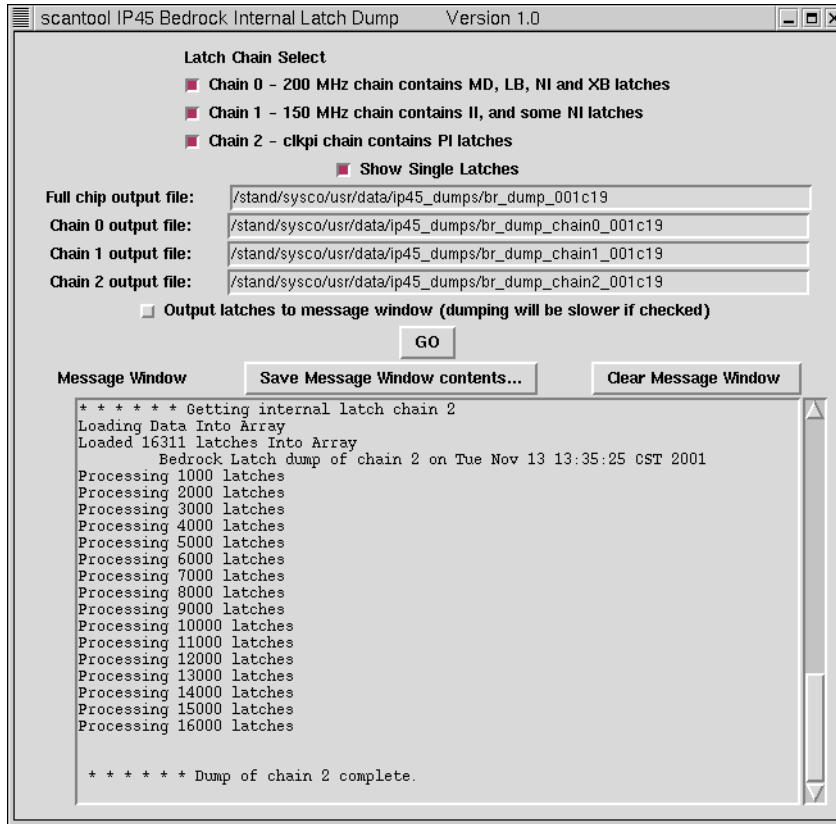


Figure 5-26 Output from the ip45_dump_latches.tk Script

15. Run the L1 reboot_11 command on the SGI Origin 300 base module.

16. Reset the system.

5.6.3 PCI Expansion Module

To dump the internal Bedrock latches from the Bedrock ASIC on a Bedrock Interface Card (BIC) in an SGI Origin 300 PCI expansion module, perform the following procedure:

1. Connect a USB cable from the L3 system controller to the SGI Origin 300 base module.
2. Log into the L3 system controller as `sgidiag` (default password: `sgi!diag`).
3. Enter the `cd /stand/sysco/usr` command to change to the `/stand/sysco/usr` directory.
4. Enter the following command to connect to the L2 emulator.

```
/stand/sysco/bin/l2
```

5. Enter the following command to start the `scantool` application:

```
/stand/sysco/bin/scantool --sysname <system_name>
```

Example: `/stand/sysco/bin/scantool --sysname lefty`

Figure 5-27 shows the `scantool` graphical user interface (also called the graphical console).

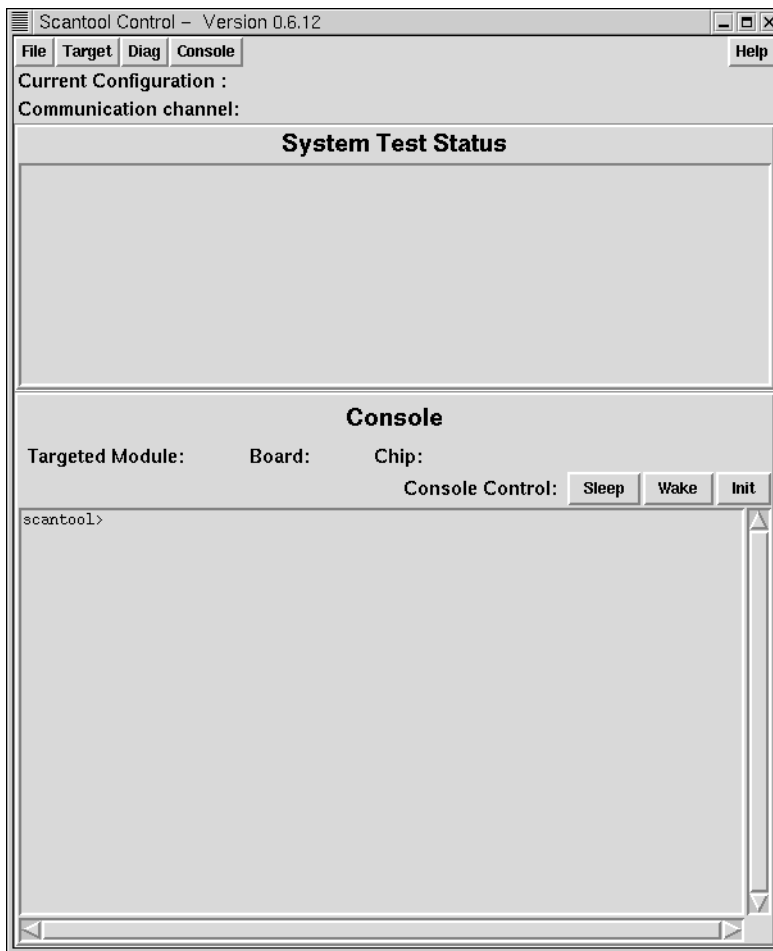


Figure 5-27 scantool Graphical User Interface (Initial Window)

6. Choose **File** -> **Set SINGLE brick Config.**

The interface displays the **Configure One Brick** window. (Refer to Figure 5-28.)

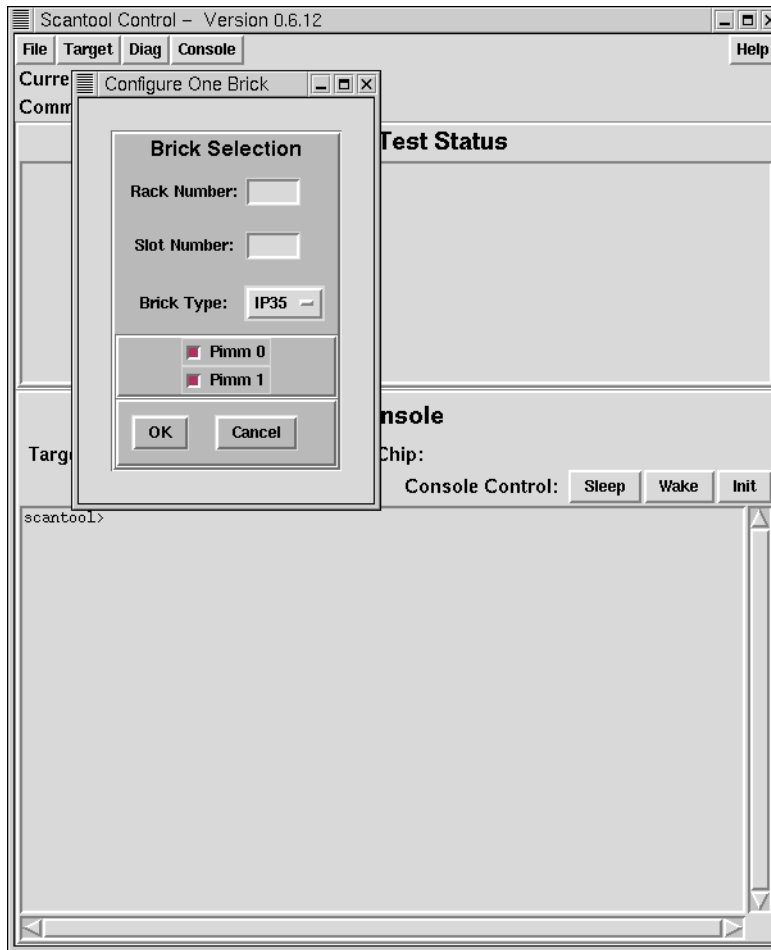


Figure 5-28 Configure One Brick Window

7. Specify the SGI Origin 300 PCI expansion module that you want to use (refer again to Figure 5-28):

- Enter the rack number in the `Rack Number` field.
- Enter the slot number in the `Slot Number` field.

If you do not know the rack and slot locations of the brick that you want to test, use the `L2 cfg` command to determine the bricks that are available.

- Select `PE` from the `Brick Type` menu.

Figure 5-29 shows example settings used to select the PCI expansion module located in slot 15 of rack 1.

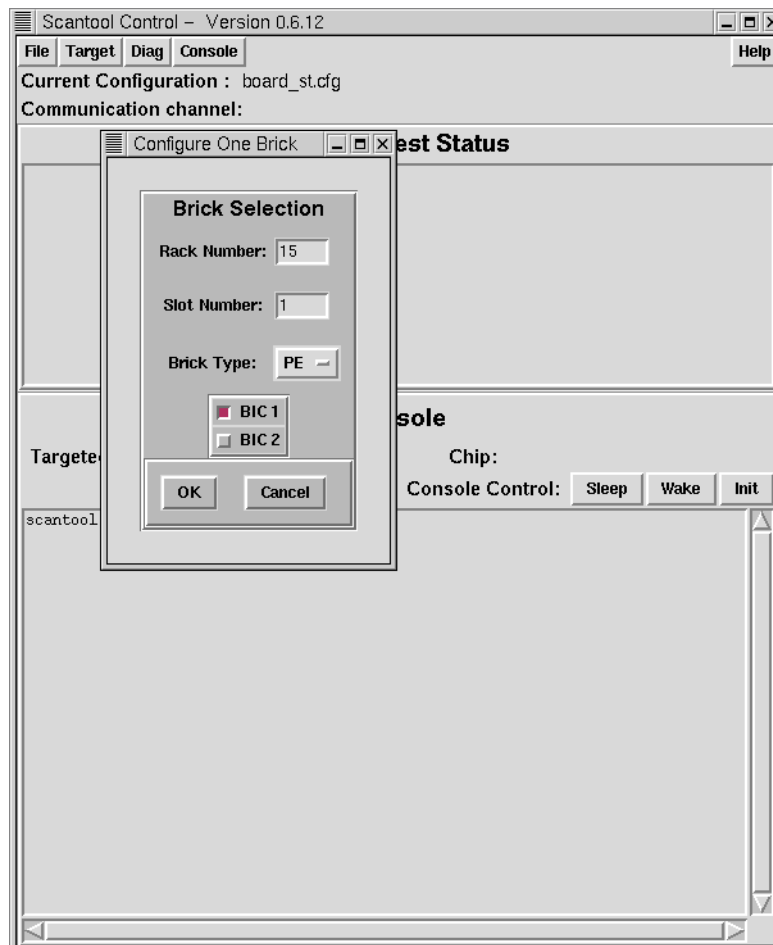


Figure 5-29 Choosing the PCI Expansion Module to Use

8. Click on the OK button.
9. Choose **Diag** -> **Start Console Diag**.
10. Double-click on pebrick.
11. Double-click on pe_dump_br_latches.tk. (Refer to Figure 5-30.)

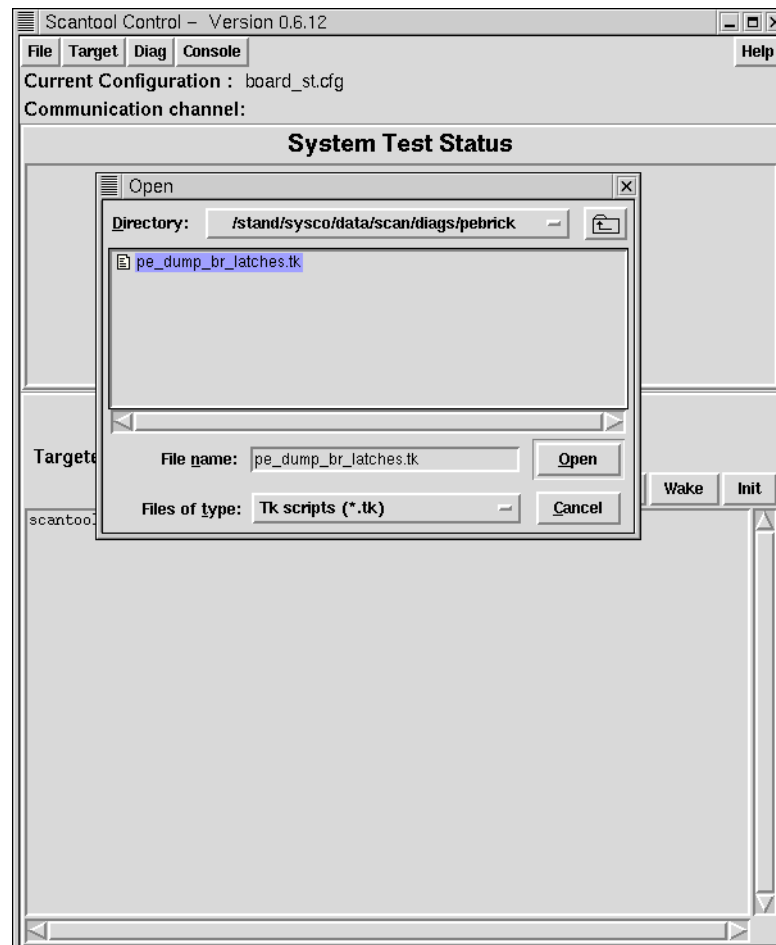


Figure 5-30 Selecting the PCI Expansion Module Script to Run

The interface displays the PE Brick Bedrock Internal Latch Dump window. (Refer to Figure 5-31.)

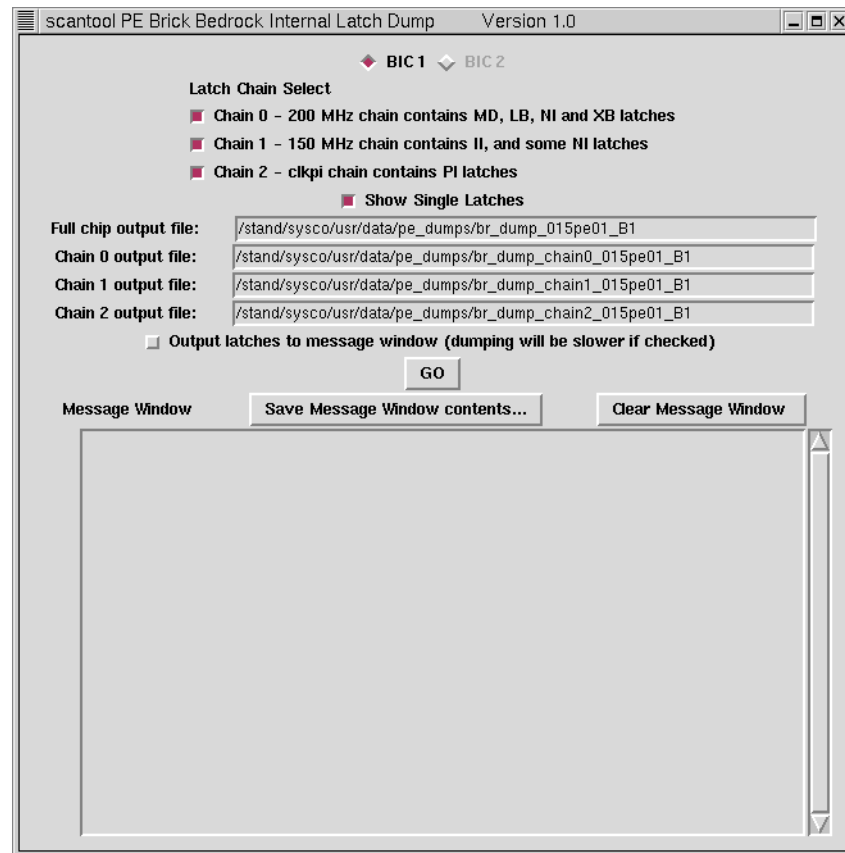


Figure 5-31 PE Brick Bedrock Internal Latch Dump Window

12. Modify the parameters, if necessary.

The parameters select the BIC to use (only BIC 0 is available in PCI expansion modules), the scan chains to use, the output files to use, and whether or not `scantool` should show single latches and output the latches to the `Message Window`.

13. Click on the `GO` button.

The interface displays a warning message, similar to the message displayed by the `ip45_dump_latches.tk` script. (Refer again to Figure 5-25.)

14. Click on the `OK` button.

Output appears in the lower portion of the window. (Refer to Figure 5-32.) Output is also written to the specified files.

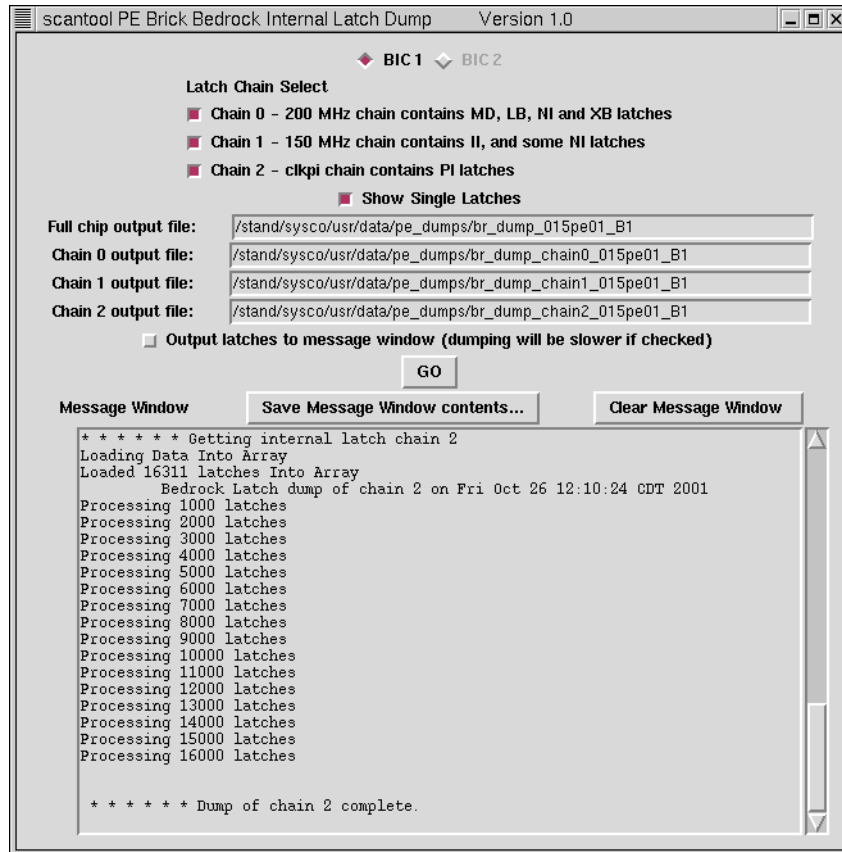


Figure 5-32 Output from the pe_dump_br_latches.tk Script

15. Run the L1 `reboot_l1` command on the PCI expansion module.
16. Reset the system.

5.7 Dumping the Internal Router Latches

If a system hangs and does not respond to an NMI, it is sometimes useful to dump the internal router latches. You may be requested to collect this data and return it to SGI for further analysis.

To dump the internal router latches from an R brick in an SGI 3000 series system, perform the following procedure:

1. Connect a USB cable from the L3 system controller to the SGI 3000 series system.
2. Log into the L3 system controller as `sgidiag` (default password: `sgi!diag`).
3. Enter the `cd /stand/sysco/usr` command to change to the `/stand/sysco/usr` directory.
4. Connect to the L2 system controller.

- To use a hardware L2 connection:

Connect the L3 system controller to the L2 system controller with an Ethernet connection by using a crossover cable.

Obtain the IP address of the L2 system controller:

```
cat /var/state/dhcp/dhcpd.leases
```

Enter the command: `/stand/sysco/l2term --l2 <host>`

Example: `/stand/sysco/l2term --l2 10.1.1.37`

- To use the L2 emulator:

Enter the command: `/stand/sysco/bin/l2`

5. Start the `scantool` application:

```
/stand/sysco/bin/scantool --sysname <system_name>
```

Example: `/stand/sysco/bin/scantool --sysname lefty`



Figure 5-33 scantool Graphical User Interface (Initial Window)

6. Choose **File** -> **Set SINGLE Brick Config.**

The interface displays the **Configure One Brick** window. (Refer to Figure 5-34.)

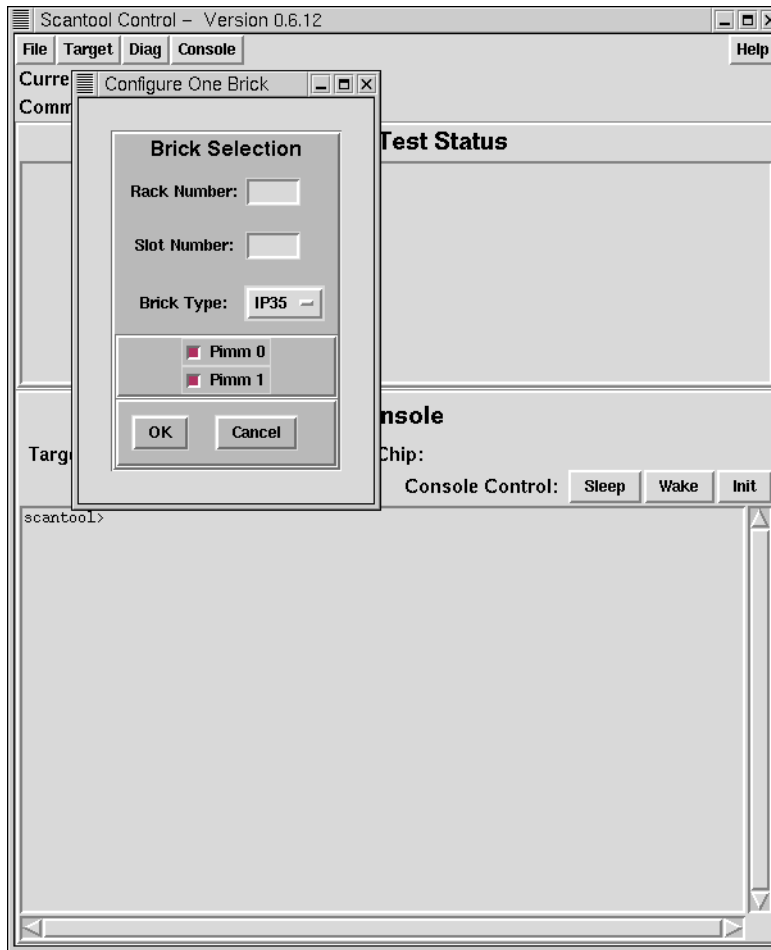


Figure 5-34 Configure One Brick Window

7. Specify the R brick that you want to use (refer again to Figure 5-34):

- Enter the rack number in the Rack Number field.
- Enter the slot number in the Slot Number field.

If you do not know the rack and slot locations of the brick that you want to test, use the `L2 cfg` command to determine the bricks that are available.

- Select R from the Brick Type menu.

Figure 5-35 shows example settings used to select the R brick located in slot 23 of rack 7.

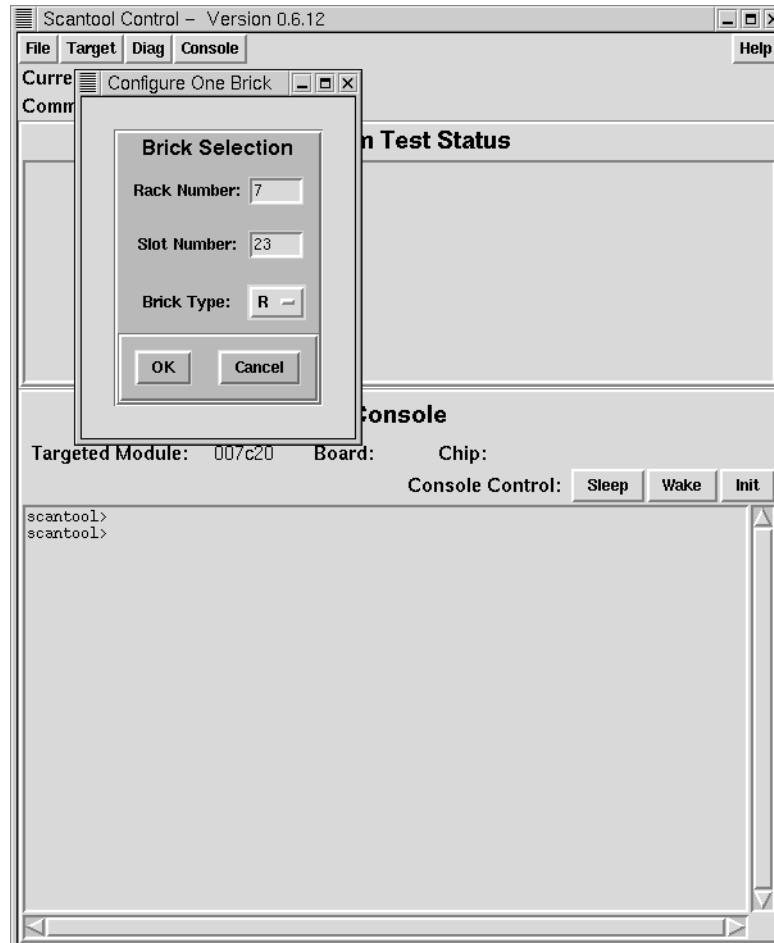


Figure 5-35 Choosing the R Brick to Use

8. Click on the OK button.
9. Choose **Diag -> Start Console Diag.**
10. Double-click on **rbrick.**
11. Choose **Tcl scripts** from the **Files of type** menu.
12. Double-click on **r_dump_latches.tcl.** (Refer to Figure 5-36.)

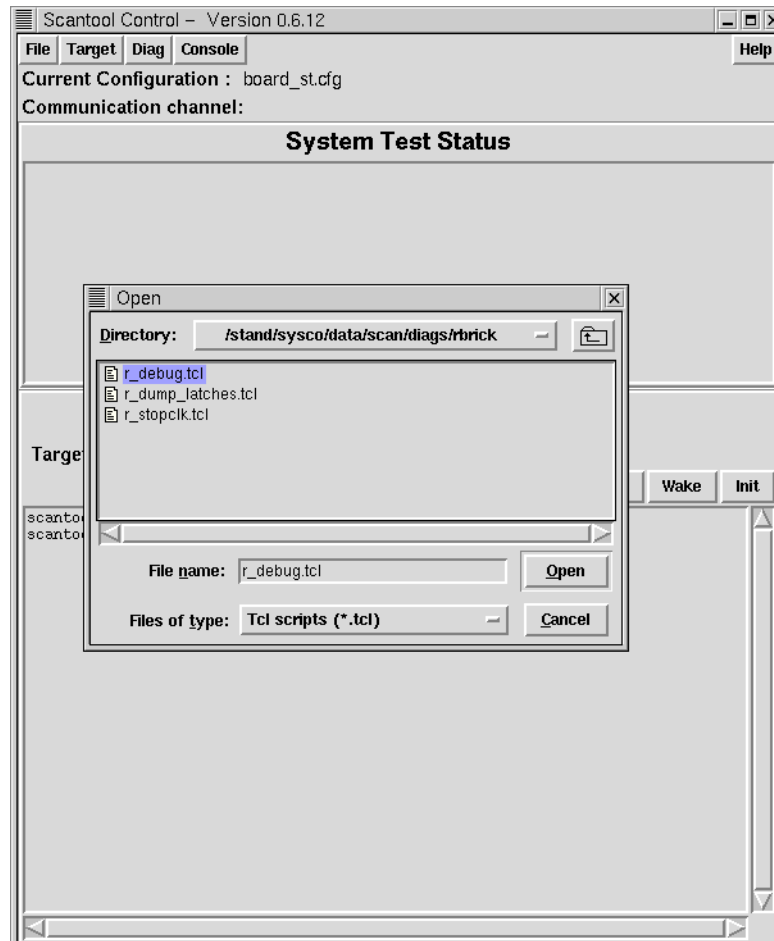


Figure 5-36 Selecting the R Brick Script to Run

The utility automatically runs for two or three minutes. Output appears in `Console` portion of the window. (Refer to Figure 5-37.)

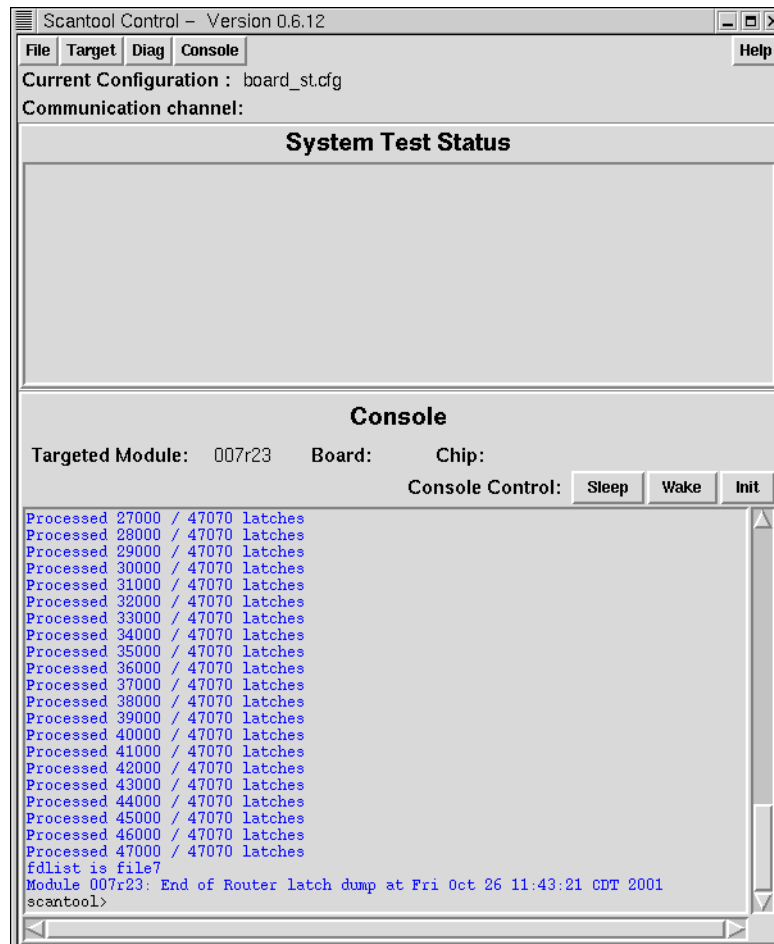


Figure 5-37 Output from the `r_dump_latches.tcl` Script

`scantool` automatically sends output to the following file:

```
/stand/sysco/usr/data/rtr_dumps/rtr_dumpRRRrSS
```

where *RRR* is the rack and *SS* is the slot of the selected R brick.

Note: The system console may show that the system reboots. If this happens, the router clock stops and discovery shows that the router being dumped is not configured.

13. Run the `L1 reboot_11` command on the selected R brick.
14. Reset the system.

Reader Comment Form

Title: Scan Tools (SGI® Origin® 300 System
and SGI® 3000 Family)

Number: 108-0263-004

Your feedback on this publication will help us provide better documentation in the future. Please take a moment to answer the few questions below.

For what purpose did you primarily use this document?

_____ Troubleshooting

_____ Tutorial or introduction

_____ Reference information

_____ Classroom use

_____ Other - please explain

Using a scale from 1 (poor) to 10 (excellent), please rate this document on the following criteria and explain your ratings:

_____ Accuracy _____

_____ Organization _____

_____ Readability _____

_____ Physical qualities (binding, printing, page layout) _____

_____ Amount of diagrams and photos _____

_____ Quality of diagrams and photos _____

Completeness (Check one and explain your answer)

_____ Too much information _____ Too little information _____ Correct amount

You may write additional comments in the space below. Mail your comments to the address below, fax them to us at +1 715 726 4353, or e-mail them to us at *spt@sgi.com*. When possible, please give specific page and paragraph references. We will respond to your comments in writing within 48 hours.

NAME _____

JOB TITLE _____

E-MAIL ADDRESS _____

SITE/LOCATION _____

TELEPHONE _____

DATE _____

[or attach your business card]



Attn: Service Publications and Training
890 Industrial Boulevard
P.O. Box 4000
Chippewa Falls, WI 54729-0078
USA

