O2_ide Reference Guide

SGI Confidential & Proprietary Information - For Internal Recipients Only

Document Number 108-0316-001

CONTRIBUTORS
Written by Darrin Goss
Edited by Cindi Leiser
Production by Rhonda Kunsman
Engineering contributions by Jun Galamay and Roberto Romano

Record of Revision

Version	Description
001	January 2001 Original printing.

Contents

Introduction

This document contains information about the `O2_ide` service tool for Silicon Graphics O2 visual workstations. This information includes a general product overview; instructions on installation, configuration, and operation of this program; and descriptions of the test suite.

The following typographic conventions are used throughout this document:

Convention	Meaning
`TYPEWRITER FONT`	Denotes literal items such as command names, file names, routines, directory names, path names, signals, messages, and programming language structures.
italic font	Denotes variable entries and words or concepts being defined.
`bold typewriter font`	In screen drawings of interactive sessions, denotes literal items entered by the user. Output is shown in nonbold typewriter font.
[]	Indicates an optional item.
<>	Indicates a required variable within an optional item.

Chapter 1

Introduction to O2_ide

This chapter provides an overview of `O2_ide` and lists the diagnostics that you can use to troubleshoot failures on Silicon Graphics O2 visual workstations.

1.1 Product Overview

`O2_ide` is a standalone diagnostic environment that is used to diagnose failing hardware components on Silicon Graphics O2 visual workstations. It enables you to run diagnostic tests in an environment that cannot be accessed from a user program that is running under the IRIX operating system.

`O2_ide` diagnostics require a dedicated system; you must reboot the system to run `O2_ide`. In other words, the IRIX operating system cannot be running when you use `O2_ide` diagnostics. You should use `O2_ide` diagnostics to diagnose failures when IRIX based diagnostics do not provide sufficient error information or to isolate failing hardware that IRIX based diagnostics cannot test.

This document describes `O2_ide` version 6.5.

1.2 Installing and Configuring O2_ide

Use the following procedure to install and configure `O2_ide`.

1. The *Internal Support Tools 2.3* CD release includes version 6.5 of `O2_ide`. Refer to the instructions that are included in the release package for information about how to install the `O2_ide` software in the `/stand` directory on your system disk.
2. Enter the following command to load `O2_ide`:

   ```
   >>boot dksc(x,y,z)/stand/O2_ide
   ```

   where *x*, *y*, and *z* refer to the controller, disk, and slice number of the disk where `O2_ide` is installed.

After you boot `O2_ide` from the Command Monitor (that is, at the BaseIO command prompt, `>>`), the `ide>>` prompt appears. At this prompt, you can enter commands that control `O2_ide` and commands that run diagnostic tests.

Future updates to `O2_ide` diagnostics will be available at the Internal Support Tools group Web site or on future CD releases. Refer to the following URL:

http://ist.csd.sgi.com/Tools/Home_pages/products_main.html

Chapter 2

Commands that Control O2_ide

This chapter lists the commands that you can use to control `O2_ide`. Although there are many commands and diagnostics that are available within `O2_ide`, you should use only the commands that are documented in this manual. Some diagnostics have prerequisite tests, commands, and environmental variables that must be set before they can run correctly. If you run the undocumented tests without proper initialization, you may leave `O2_ide` in an unknown state or cause an `O2_ide` exception.

2.1 version Command

The `version` command displays the current version number of `O2_ide` and its image build date. This command uses the following syntax:

```
version
```

2.2 report Command

The `report` command sets the verbosity level (1 through 4) of the `O2_ide` diagnostics. This command uses the following syntax:

`report=<`*value*`>`

2.3 hinv Command

The `hinv` command displays the contents of the system hardware inventory table. This table is created each time the system is booted, and it contains entries that describe the various hardware components in the system. The items in the table include the main memory size, cache sizes, floating-point unit, and disk drives. Without arguments, the `hinv` command displays a one-line description of each entry in the table. This command uses the following syntax:

```
hinv
```

2.4 printenv Command

The `printenv` command displays all current NVRAM variable settings if you enter it without arguments, or it displays the value of a single specified NVRAM variable. This command uses the following syntax:

```
printenv
```

or

```
printenv <variable>
```

2.5 setenv Command

The `setenv` command sets a selected variable that is stored in NVRAM to a specified value. This command uses the following syntax:

```
setenv <variable> <value>
```

2.6 unsetenv Command

The `unsetenv` command clears a selected variable that is stored in NVRAM. This command uses the following syntax:

```
unsetenv <variable>
```

2.7 exit and quit Commands

The `exit` and `quit` commands exit the `O2_ide` environment and return you to the `System Maintenance Menu`. These commands use the following syntax:

```
exit
```

or

```
quit
```

Chapter 3

Diagnostic Tests

This chapter lists the `O2_ide` diagnostic tests and describes how to run them and the output that they generate.

3.1 Available Diagnostic Tests

`O2_ide` diagnostics are directed tests; each test focuses on a specific area of the system. Table 3-1 lists the diagnostic tests that are available under `O2_ide`.

Table 3-1 Diagnostic Tests by Area Tested

Area	Test Name	Description
CPU	`fpu`	Floating-point unit (FPU) test
	`tlb`	Translation lookaside buffer (TLB) test
	`dcache1`	Data cache parity test
	`icache1`	Instruction cache data parity test
	`icache2`	Instruction cache miscellaneous test
	`scache1`	Secondary cache address test
	`scache2`	Secondary cache address and data test
	`scache3`	Secondary cache page invalidate test
	`scache4`	Secondary cache random line miss test
	`scache5`	Secondary cache address and data test
	`scache6`	Secondary cache address test
	`tcache1`	Tertiary cache address and data test

Table 3-1 (continued) Diagnostic Tests by Area Tested

Area	Test Name	Description
Memory	`memtest`	CPU memory test (original)
	`memtest2`	CPU memory test (new)
	`memtest3`	Fast 64-bit address and data memory test
I/O	`scsi`	Basic SCSI test
	`pci`	PCI configuration space address and data test
	`xsci`	SCSI disk exerciser (exercises all SCSI disk drives that are found)
	`plpreg`	Parallel port register test
	`plploop`	Parallel port loopback test
	`plp_ext_loop`	Parallel port external loopback test
	`serialloop`	Serial port loopback test
	`maceec`	MACE Ethernet hardware test
	`maceecs`	MACE Ethernet hardware stress test
Graphics	`gfx_reg_test`	Graphics registers test
	`qnd_gfx_test`	Quick and dirty graphics test
Other	`clock`	Real-time clock test
	`cntrs`	UST counter and compare register interrupts test
	`i2c_reg`	MACE I2C register test

3.2 Running Diagnostic Tests

To run a test, enter the test name (as shown in the “Test Name” column of Table 3-1) at the `ide>>` prompt.

3.3 Test Output

As a test runs, it returns output to the console.

For example:

```
ide>> tlb
Translation Lookaside Buffer (TLB) test
TLB data test
TLB probe test
TLB translation test
TLB valid bit test
TLB mod bit test
TLB pid test
TLB global bit test
TLB cached bit test
Test completed with no errors.
```

Note: The amount of output that a test displays depends on the verbosity level that you select with the `report` command. SGI recommends that you select the maximum verbosity level before you run any tests (enter **`report=4`**).

3.3.1 Passing Output

When a diagnostic test completes executing without detecting errors, it displays the following message:

```
Test completed with no errors.
```

For example:

```
ide>> dcache1
Data cache data parity test
Test completed with no errors.
```

3.3.2 Failing Output

When a diagnostic test detects an error, it displays the text `- ERROR -` and then provides more information about the error.

For example:

```
ide>> serialloop
DMA loopback test, Serial B
Error code = IP32.08.0e.01.02, w1 = 00000000, w2 = 00000000
                                                          - ERROR -
ERROR: no rx data
Error code = IP32.08.0e.01.07, w1 = 00000000, w2 = 00000000
                                                          - ERROR -
ERROR: no interrupts
```

Reader Comment Form

Title: O2_ide Reference Guide **Number: 108-0316-001**

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