FRU Analyzer Reference Guide

Document Number 108-0166-002

Contributors

Written by Greg Russell
Illustrated by Greg Russell
Edited by Cindi Leiser
Production by Cindy Stief
Engineering contributions by Sasha Robinson and Vijay Chander

© Copyright 1997, Silicon Graphics, Inc.— All Rights Reserved

This document contains proprietary and confidential information of Silicon Graphics, Inc. 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.

Restricted Rights Legend

Use, duplication, or disclosure of the technical data contained in this document by the Government is subject to restrictions as set forth in subdivision (c) (1) (ii) of the Rights in Technical Data and Computer Software clause at DFARS 52.227-7013 and / or in similar or successor clauses in the FAR, or in the DOD or NASA FAR Supplement. Unpublished rights reserved under the Copyright Laws of the United States. Contractor / manufacturer is Silicon Graphics, Inc., 2011 N. Shoreline Blvd., Mountain View, CA 94043-1389.

Silicon Graphics, Inc.
Mountain View, California

Silicon Graphics and the Silicon Graphics logo are registered trademarks of Silicon Graphics, Inc. IRIX, Onyx2, Origin, Origin200, and Origin2000 are trademarks of Silicon Graphics, Inc. CrayLink is a trademark of Cray Research, Inc.

Contents

Introduction

This document contains information about the FRU Analyzer, a tool that helps the support provider to isolate system problems.

The FRU Analyzer manual guides the reader from an introduction to the product to an understanding of the program's output messages, including many specific examples. Appendix A lists all the conditions and conclusions that compose the analyzer portion of this tool.

The reader audience includes engineers, analysts, support personnel, and field service personnel.

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.

Within this document, reference may be made to the online man pages available under IRIX™ through the `man` command. A *man page* is a discussion of a particular element of the IRIX™ operating system or a compatible product.

Each man page includes a general description of one or more commands, routines, or other topics and provides details of their usage (command syntax, routine parameters, system call arguments, and so on). If more than one topic appears on a page, the entry will appear in the printed manual alphabetized only under its major name. You can access a man page named `ls` on-line by entering **`man ls`**.

Man pages are grouped into sections numbered from 1 to 8. Each section contains entries of a particular type. Types of entries include user commands (1), administrator commands (8), system calls (2), library routines (3), file formats (5), and device descriptions (4).

Section numbers appear in parentheses after man page names. Man pages are referenced in text by entry name and section number.

Chapter 1

Introduction to the FRU Analyzer

This Chapter provides an overview of the IRIX™ FRU Analyzer Tool (FRU).

1.1 Product Overview

The FRU Analyzer is a PROM-resident IRIX based software tool that provides the SSE with a starting point in the resolution of a system malfunction. When the operating system determines that a fatal error has occurred, the kernel process automatically collects and passes the hardware error state to the FRU Analyzer for interpretation. The FRU Analyzer applies a set of fixed rules to the hardware state and from those rules attempts to determine the cause of the fatal error. User intervention is not required.

In releases of the FRU Analyzer before version 6.4, the FRU Analyzer provides board-level replacement information relative to a system core dump. The `savecore` script automatically activates FRU at startup when the system reboots after a core dump. FRU analyzes the `unix` and `vmcore` files if they were successfully created by `savecore`. If `unix` and `vmcore` files do not exist because `savecore` was disabled at startup by `chkconfig`, FRU analyzes the `vmcore` image located in the dump device.

Beginning with release 6.4, the FRU Analyzer does not use core files and is not part of ICRASH. Origin™ systems running the IRIX 6.4 operating system provide FRU Analyzer software that resides entirely within the IRIX kernel.

FRU evaluates which boards, if any, need to be replaced and assigns a percentage confidence level to each of its evaluations. Following this analysis, the FRU Analyzer reports failure information about the boards that scored the highest probability of failure. If the hardware error state indicates more than four faults, the FRU Analyzer determines that the hardware error state is inconclusive and does not issue a report.

1.2 Product Limitations

Hardware debugging is an inherently complex task. The hardware error state is a limited set of information that is possibly inaccurate. Because of these issues, the FRU Analyzer should not be expected to identify all possible failures correctly. It is best at finding straightforward errors such as a failing memory DIMM, and it is worst at isolating complicated CrayLink™ communications infrastructure or I/O path failures that can sometimes cause misleading error states. As with all static sets of rules placed in real world environments, you should consider the conclusions of the FRU Analyzer a guide to the solution of hardware problems, but you should always interpret the results with the knowledge that they will not always be correct.

1.3 Output

The analysis of the FRU Analyzer is printed to the console in the following format:

```
*FRU ANALYSIS BEGIN


FRU ANALYSIS SUMMARY
/hw/module/1/slot/n3/node/memory: 70%
/hw/module/1/slot/n3/node/memory/bank/0: 70%


*FRU ANALYSIS END
```

When the analyzer applies its rules, it generates a list of conclusions. Each conclusion comprises a suspected hardware unit and a corresponding percentage of confidence. To avoid forcing the user to interpret multiple results, the program outputs to the console only the conclusions with the highest certainty. (All conclusions with equal certainty are printed.) At the same time, the entire analysis is stored as part of the crash dump so that more information is available if the primary conclusion is determined to be inaccurate (for example, the same failure signature and the same analyzer conclusion reoccur after the suspected FRU has been replaced).

The displayed percentage of confidence suggests how conclusively the error state relates to a failing unit. A displayed percentage value is assigned independently to each failing unit and is immediately obvious in the output message. The percentage values simply indicate the relative confidence level of each of the findings for a series of reported failing devices, and they do not necessarily total 100%.

1.4 Node Analysis

The previous example of the analysis summary demonstrates that the FRU Analyzer output must be interpreted; it is not meant to be accepted literally. The summary reports two errors of equal weight. The first conclusion points to a memory error on the IP27 board, which is located in slot 3 of module 1. The error state apparently provides insufficient

evidence to isolate the failure to a specific DIMM bank. The second conclusion specifies the bank, namely bank 0, on the same IP27 board. Both results in the example are reasonably certain, but because the first determination does not exclude the second, you should first test DIMM bank 0 to determine whether it is malfunctioning. If you determine that DIMM bank 0 is working correctly, you should then perform a more general memory test to locate the failing part. If the reported units had been unrelated and of equal weight, then either could be the cause of failure, and both should be investigated.

A percentage reported in the range of 60% to 90% indicates that the error state isolated the failure to a particular device. (Note however, that a single example is neither conclusive nor exclusive. Because of the distributive nature of the analysis, a first conclusion does not preclude a second possible cause of the failure. In other words, the FRU Analyzer can simultaneously report two different devices, each with a 60% to 90% confidence.) If the percentage is less than 60%, the error state implies that the named device is a possible failure source. A percentage of 95% means that a special rule has been triggered; this high percentage is reported for particular hardware failure symptoms that are directly attributable to a single piece of hardware (refer to Section 2.2.1, “XBOW”).

Chapter 2

FRU Analyzer Output Messages

This Chapter is a general guide to the FRU Analyzer output messages. It lists each detectable failing unit and provides a sample output message for that unit. For clarity, each FRU uses a hardware path name. (Obviously, the path names for real errors vary, depending on the location of the malfunctioning hardware. The percentage values, reported in the examples as *XX*%, are also variable.) Because the FRU Analyzer points to some items that are not included in the graph (such as sections of the HUB chip or the various CPU caches), some output messages contain final elements that do not appear in the graph.

2.1 Node Analysis

Node analysis messages report failures in the software, the IP27, the memory, and the router link.

2.1.1 Software

`Software :`*`XX`*`%`

This message indicates that the fatal error was probably caused by system software.

2.1.2 IP27

`/hw/module/1/slot/n1/node :XX%`

This example is a general error message indicating that a failure occurred somewhere on the IP27 board, but there is not enough information to isolate this failure further.

If the failure had been better defined, the error message might have pointed to one of the following IP27 board subunits:

```
/hw/module/1/slot/n1/node/himn
/hw/module/1/slot/n1/node/cpu/a
/hw/module/1/slot/n1/node/cpu/b
/hw/module/1/slot/n1/node/cpu/[a or b]/dcache
/hw/module/1/slot/n1/node/cpu/[a or b]/icache
/hw/module/1/slot/n1/node/cpu/[a or b]/scache
/hw/module/1/slot/n1/node/hub
/hw/module/1/slot/n1/node/hub/md
/hw/module/1/slot/n1/node/hub/pi
/hw/module/1/slot/n1/node/hub/ni
/hw/module/1/slot/n1/node/hub/ii
```

Note: The FRU Analyzer seldom reports hub failures because most hub failures are detected by diagnostics during the boot process.

2.1.3 Memory

`/hw/module/1/slot/n1/node/memory :`*`XX`*`%`

This example indicates a general memory failure that was not isolated to a specific DIMM. Most memory error messages provide a DIMM number and appear as:

`/hw/module/1/slot/n1/node/memory/bank/0 :`*`XX`*`%`

2.1.4 Router Link

`/hw/module/1/slot/r2/router/link/0 :`*`XX`*`%`

This message indicates that the router has received corrupted packets from a link (in this example, link 0). The FRU Analyzer cannot effectively isolate CrayLink network errors because although the failure symptoms are readily detectable, the cause of the failure is usually obscure. For example, the error message in the preceding example could have resulted from any of the following failures:

- the cable connected to this link is bad or has a bad connection
- the transmitting router is failing and sending corrupted information
- the receiving router is failing and is erroneously reporting corrupted packets

When a single cause of failure is not apparent, you must investigate all possible causes.

2.2 I/O Analysis

The MR release of the IRIX 6.4 operating system does not perform I/O analyses. Later releases will report messages similar to those that follow.

2.2.1 XBOW

`/hw/module/1/slot/n1/node/xtalk/0/xbow :`*`XX`*`%`

This message indicates a malfunction in the crossbow ASIC. Crossbow failures are improbable; you are more likely to encounter a link or widget problem.

One known crossbow hardware problem, however, triggers a special rule and displays the following message:

`/hw/module/1/slot/n1/node/xtalk/0/xbow :95%`

This message indicates that the crossbow circuit should be replaced. Currently, this rule is the only rule with a 95% confidence rating.

2.2.2 XBOW Link that is Corrupting Data

A failure of the crossbow link results in the following message:

`/hw/module/1/slot/io5/basio/link :`*`XX`*`%`

This general error message indicates that the crossbow circuit has received corrupted data from a link. Either the attached widget has failed and is transmitting corrupted data, or a bad connection exists between the widget board and the midplane.

2.2.3 I/O Boards

All of the following messages indicate that the I/O board in the named slot is malfunctioning:

`/hw/module/1/slot/io1/baseio :`*`XX`*`%`

`/hw/module/1/slot/io1/baseio/bridge :`*`XX`*`%`

`/hw/module/1/slot/io1/baseio/bridge/ssram :`*`XX`*`%`

In these three examples, each consecutive message provides more specific information about the problem than the preceding messages. The second and third messages report a bridge chip failure. The third case isolates the failure to the SSRAM on the bridge chip.

PCI devices that are connected to the BaseIO card may also malfunction. Conclusions for this type of error appear in one of the following forms:

```
/hw/module/1/slot/io1/baseio/pci/1 :XX%
/hw/module/1/slot/io1/mscsi :XX%
/hw/module/1/slot/io1/menet :XX%
/hw/module/1/slot/io1/graphics :XX%
/hw/module/1/slot/io1/fibre_channel :XX%
/hw/module/1/slot/io1/pci_xio :XX%
```

The FRU Analyzer can also report failures for devices that are not included in the I/O list. These failures appear as:

```
/hw/module/1/slot/io1/unknown
```

Appendix A

FRU Analyzer Rules

This Appendix lists all the conditions and conclusions for FRU Analyzer, version 1.6. The Section column indicates the hardware area within which the hardware state registers are located. The Condition column contains the error state that defines the rule. The FRU column indicates the suspected unit. The Conf column contains the percentage of confidence assigned to the diagnoses. The Type column points to the location of the rule in the FRU Analyzer program (coded rules are located in a `*.c` file whereas tabular rules are parsed from the file `sn0_fru_rules.txt`.)

Table A-1 Conditions and Conclusions for FRU Analyzer

Section	Condition	FRU	Conf	Type
Router				
	LINK STATUS bits 24..33 set	Link	60%	Coded
	LINK STATUS bit 32< Illegal port>	Link	70%	Coded
PI-hub				
	ERR_INT_PENDbit 23<CPUA SysState Tag error> OR bit17<CPUA Syscmd parity err during addr cycle> OR bit15<CPUA Syscmd parity err during data cycle> OR bit13<CPUA SysAD error during addr cycle> OR bit 11<CPUA SysAD error during data cycle> OR bit9<CPUA Sysstate Parity Error>	CPU0 Hub	90% 80%	Tabular
	ERR_INT_PENDbit 22<CPUB SysState Tag error> OR bit16<CPUB Syscmd parity err during addr cycle> OR bit14<CPUB Syscmd parity err during data cycle> OR bit12<CPUBSysAD error during addr cycle> OR bit 10<CPUB SysAD error during data cycle> OR bit8<CPUB Sysstate Parity Error>	CPU1 Hub	90% 80%	Tabular
	ERR_INT_PENDbit5<CPUA WRB TERR> OR bit4 <CPUB WRB TERR>	Hub	70%	Tabular
	ERR_INT_PENDbit6<CPUA WRB WERR> OR bit7<CPUB WRB WERR>	Mem	70%	Tabular

Table A-1 (continued) Conditions and Conclusions for FRU Analyzer

Section	Condition	FRU	Conf	Type
	(ERR_STATUS_0_A bit63<VALID> AND ERR_STATUS_0_A bits23..22<TYPE> = 0<WERR> AND ERR_STATUS_1_A bit42<RRB / WRB> = 1) OR (ERR_STATUS_0_B bit63<VALID> AND ERR_STATUS_0_B bits23..22<TYPE> =0<WERR> AND ERR_STATUS_1_Bbit42<RRB / WRB> = 1)	Bank#	70%	Coded
	(ERR_STATUS_0_A bit63<VALID> AND ERR_STATUS_0_A bits23..22<TYPE> = 1<UPRW> OR (ERR_STATUS_0_B bit63<VALID> AND ERR_STATUS_0_B bits23..22<TYPE> = 1<UPRW>)	Bank #	70%	Coded
	(ERR_STATUS_0_A bit63<VALID> AND ERR_STATUS_0_A bits23..22<TYPE> = 2<DERR>	Bank#	70%	Coded
	(ERR_STATUS_0_A bit63<VALID> AND ERR_STATUS_0_A bits23..22<TYPE> = 3<TOUT>	Hub	70%	Coded
	(Err_status_0_a bit63<Valid> AND (Err_status_1_a bit 44<T> OR Err_status_1_a bit 48<W> OR (bit42<RRB / WRB> = 1 AND bit45<I>)) OR (Err_status_0_b bit63<Valid> AND (Err_status_1_b bit 44<T> OR Err_status_1_b bit 48<W> OR (bit42<RRB / WRB> = 1 AND bit45<I>)) OR	Mem	70%	Coded
MD-hub				
	MD_PROTOCOL_ERROR bit63 <VALID>	Software	60%	Tabular
	MD_MISC_ERROR bit 9 <ILL_MSG> OR bit7<ILL_REV> OR bit5<LONG_PACK> OR bit3<SHORT_PACK>	Hub	70%	Tabular
	Dir Error bit63<UCE_VALID> OR bit62<AE_VALID>	Mem Software	80% 40%	Coded
	Mem Error bit63<UCE_VALID> AND Syndrome<> 0xff	Bank#	90%	Coded
	Mem Error bit63<UCE_VALID> AND Syndrome = 0xff	Mem Bank#	50% 50%	Coded

Table A-1 (continued) Conditions and Conclusions for FRU Analyzer

Section	Condition	FRU	Conf	Type
II-hub				
	WIDGET_STATUS bit32 <CRAZY>	Midplane connection	80%	Coded
	IO CRB Entry 0_A bits55..52 <ERRCODE> = 1<PERR> OR IO CRB Entry 0_A bits55..52 <ERRCODE> = 2 <WERR> OR IO CRB Entry 0_A bits55..52 <ERRCODE> = 3 <AERR> OR IO CRB Entry 0_A bits55..52 <ERRCODE> = 4<PWERR> OR IO CRB Entry 0_A bits55..52 <ERRCODE> = 5<PWERR>	Software	70%	Coded
	IO CRB Entry 0_A bits55..52 <ERRCODE> = 8<DERR>	MD	60%	Coded
NI-hub				
	NI_PORT_ERROR bit 36<INTERNAL ERROR>	Hub PI MD II	90% 60% 60% 60%	Tabular
	NI_VECTOR_STATUS bits2..0 <TYPE> = 4<ADDR ERR> OR 5<CMD ERR> OR 6<PROT ERR>	Software	70%	Coded
CPU				
	Cache_Error bits31..30<TYPE> = 0<I_CACHE>	ICACHE	90%	Coded
	Cache_Error bits31..30<TYPE> = 1<D_CACHE>	DCACHE	90%	Coded
	Cache_Error bits31..30<TYPE> = 2<S_CACHE> AND (MD_MEM_ERROR bit 63<UCE_VALID> not set OR MD_MEM_ERROR bit 63 set but mem_error address is different from the cache error address)	SCACHE	90%	Coded
	Cache_Error bits31..30<TYPE> = 2<S_CACHE> AND (MD_MEM_ERROR bit 63<UCE_VALID> set AND MD_MEM_ERROR address = CACHE ERROR addr AND Mem Ecc = 0xff)	SCACHE	50%	Coded
	Cache_Error bits31..30<TYPE> = 3<I_CACHE> AND Cache_Error bits25..23 <Sys Intf Error>	HIMM	70%	Coded
XBOW				
	WIDGET 0 STATUS bit 5 <register access error>	Software	90%	Tabular
	WIDGET 0 STATUS bit 2 <crosstalk error> and source widget is valid	Source Widget[a] Source Link	70% 70%	Coded
	(LINK STATUS bit 16 <input over allocation> OR LINK STATUS bit 4 <LLP receiver error>) AND XBOW revision < 3	XBOW	FLAG[b]	Coded
	LINK STATUS bit 17 <illegal destination>	Link's Widget Software	70% 60%	Coded

Table A-1 (continued) Conditions and Conclusions for FRU Analyzer

Section	Condition	FRU	Conf	Type
	LINK STATUS bit 1 <max request timeout>	XBOW	90%	Coded
	LINK STATUS bit 0 <packet timeout error source>	Link's Widget Link	70% 60%	Coded
Bridge				
	INT STATUS bit 16 <SSRAM_PERR> AND RAM PERR bit 28 <SIZE_FAULT> NOT (PCI ERROR ADDR UPPER bit20 <PCI_DEV_MASTER>	Software	90%	Coded
	INT STATUS bit 16 <SSRAM_PERR> AND RAM PERR bit 28 <SIZE_FAULT> AND PCI ERROR ADDR UPPER bit 20 <PCI_DEV_MASTER>	PCI Master Dev [c]	90%	Coded
	INT STATUS bit 16 <SSRAM_PERR> AND NOT (RAM PERR bit 28 <SIZE_FAULT>) AND (any of RAM PERR bits 16-23 <PAR_BYTE>)	Bridge SSRAM	90%	Coded
	INT STATUS bit 28 <BAD_XRESP_PACKET> OR INT STATUS bit 27 <BAD_XEQ_PACKET>	Widget's Link XBOW	80% 60%	Tabular
	INT STATUS bit 24 <INVALID_ADDRESS> OR INT STATUS bit 23 <UNSUPPORTED_XOP> OR INT STATUS bit 22 <XREQ_FIFO_OFLOW>	Software	90%	Tabular
	INT STATUS bit 14 <PCI_PARITY> OR INT STATUS bit 13 <PCI SERR> OR INT STATUS bit 12 <PCI_PERR>	IO Board	80%	Coded
	PCI ERROR ADDR UPPER bit 20 <PCI_DEV_MASTER> AND (INT STATUS bit 13 <PCI_SERR> OR INT STATUS bit 12 <PCI_PERR>)	PCI Master Dev	80%	Coded
	NT STATUS bit 11 <PCI_MASTER_TOUT> OR INT STATUS bit 10 <PCI_RETRY_CNT>	PCI Master Dev	80%	Tabular
	INT STATUS bit 9 <XREAD_REQ_TOUT>	Software	90%	Tabular

a. The source widget is determined by bits 27 through 24 of the Crossbow Error Command register.

b. FRU_FLAG_CONF is a flag value that allows special output messages. It is used for exact harware failure rules.

c. The master PCI device is stored in bits 16 through 18 of the ERROR ADDR UPPER register unless bit 20 in that register is 0 (showing that the bridge itself was the PCI master).