Supplement to Sun Fire 6800/4810/4800/3800 Systems Troubleshooting Manual

Error Examples and Analysis

Sun Microsystems, Inc.
4150 Network Circle
Santa Clara, CA 95054 U.S.A.
650-960-1300

Part No. 816-5402-11
November 2002, Revision 01

Send comments about this document to: docfeedback@sun.com

Please
Recycle

Contents

Figures

Tables

Code Samples

CHAPTER 1

Introduction

This document is intended as a supplement to the existing *Sun Fire 6800/4810/4800/3800 Troubleshooting Manual* and will focus on error examples. As there are existing ITT classes covering architecture, only detail helpful in understanding errors will be reviewed. Please refer to the references listed herein for more detailed architectural information.

1.1 Scope of Coverage

This document is organized as follows:

Chapter 1: provides useful background information pertaining to how errors are logged and what information error messages contain.

Chapter 2: contains information, examples and analysis for CPU/Schizo and CPU/Schizo related errors

Chapter 3: contains information, examples and analysis for Data Path errors reported by the SC

Chapter 4: comtains examples and analysis for Domain System errors reported by the SC

Chapter 5: contains examples and analysis for environmental errors

1.2 Overview

There are two broad classifications of error messages, errors from the SC, which has a view of the system address, control and data transfer ASICs. And those output by Solaris which has a view of the CPUs/Schizos and CPU related errors such as Memory, Ecache, Dcache, Pcache etc.

Note – Interconnect errors (L1-L1 and L1-L2) are detailed in Appendix E of the *Sun Fire 6800/4810/4800/3800 Systems Troubleshooting Manaul.*

1.2.1 Solaris Error Detection

The following errors can be output from the Solaris software:

- CPU
  - ECC (IVC/IVU/EDU/UE/CE/CPC/CPU)
  - Ecache Tag (TUE/TSCE/THCE)
  - Dcache Tag (DDSPE/DISPE)
  - Icache Tag (IDSPE/ITSPE)
  - Pcache
  - Mtag (EMC/EMU)
  - Timeout (TO/DTO)
  - Bus (BERR/DBERR)
  - Protocol (PERR), also results in Domain SYSTEM ERROR
  - Internal IERR), also results in Domain SYSTEM ERROR
  - ISAP (Incoming System Address Parity Error), also results in Domain SYSTEM ERROR
  - SafErr
- Schizo
  - ECC (UE/CE)
  - Safari-side (repeater/centerplane)
  - PCI-side
  - ISAP (Incoming System Address Parity Error), also results in Domain SYSTEM ERROR

See Chapter 2 for more information on these errors.

1.2.2 System Controller Error Detection

The following errors can be output from the SC-App software:

- ECC (L1/L2 DXs)
- Domain SYSTEM ERRORs
  - ConsoleBus
  - SafariPort
  - AR
  - SDC
  - DX
  - SBBC
  - System clock
  - Parity
  - Timeouts
  - Protocols

Refer to Chapter 3 and Chapter 4 for more information these errors.

1.3 How to Use this Document

Once you are confronted with an error message, and have determined the source, CPU/Schizo and CPU/Schizo related (Solaris messaging) or, data path and domain SYSTEM ERRORs (SC messaging), proceed to the applicable chapter as follows:

- Chapter 2: errors reported by CPUs/Schizos
- Chapter 3: data path errors reported by the SC
- Chapter 4: domain SYSTEM ERRORs reported by the SC
- Chapter 5: environmental errors

Find an example error that is similar and then read the entire section as it will cover related errors that you may need to understand in order to interpret your example. If you can't find anything similar to your example, please send the message to me at:

`serengeti-support@sun`

You can also use TABLE 1-1 to match a unique identifier within the body af an error message to similar messages and/or code examples.

TABLE 1-1 Mapping of Key Error Message Identifiers To Code Examples

Identifier	Description	Code example
CPU/Schizo and CPU/Schizo Related Errors		
BERR/DBERR	Bus errors	CODE EXAMPLE 2-9
CPC	Corectable ECC error found while CPU is supplying data out of it's cache to another CPU	
CPU	Uncorrectable ECC error found while CPU is supplying data out of it's cache to another CPU	
DDSPE	Primary Data Cache data or tag parity error	CODE EXAMPLE 2-4
DTSPE	Primary Data Cache tag parity error	CODE EXAMPLE 2-5
EDU	Uncorrectable ECC error found while CPU is supplying data out of it's cache to another CPU	
EDU:BLD	Uncorrectable ECC error found while CPU is supplying data out of it's cache to another CPU	
EMC	Hardware corrected system bus Mtag ECC error	
EMU	Uncorrectable, multi-bit system bus Mtag ECC error	
ErrorStatus SBBC-SafErr	CPU/Schizo device asserted an error (also causes domain SYSTEM ERROR)	CODE EXAMPLE 2-20
IDSPE	Primary Instruction Cache data parity error	CODE EXAMPLE 2-6
ITSPE	Primary Instruction Cache tag parity error	CODE EXAMPLE 2-7
PCI excessive retry	PCI excessive retry	CODE EXAMPLE 2-21
THCE	Hardware corrected, single-bit External Cache tag ECC error	CODE EXAMPLE 2-3
TO/DTO	Timeout errors	CODE EXAMPLE 2-8
TSCE	Software corrected, single-bit External Cache tag ECC error	CODE EXAMPLE 2-2
TUE	Uncorrectable, multi-bit External Cache tag ECC error	
WDC	Corectable ECC error found while CPU is pushing data out of it's cache	
WDU	Uncorrectable ECC error found while CPU is pushing data out of it's cache	
For ECC syndromes other than those listed see TABLE 2-4 in Chapter 2		

TABLE 1-1 Mapping of Key Error Message Identifiers To Code Examples *(Continued) (Continued)*

Identifier	Description	Code example
0x003 (ECC Syndrome)	Generated by a CPU as the result of store-merge operation to a line that already has an uncorrectable ECC in it's Ecache.	
0x003 (ECC Syndrome)	Generated by a Schizo as the result of a partial-line DMA writes to a line that was previously read from the system bus with UE ECC	
0x071 (ECC Syndrome)	Generated by a CPU as the result of a writeback or copyout of a line that previously contained UE ECC	
0x11c (ECC Syndrome)	Generated by a CPU as the result of a Bus Error	
Domain System Errors		
`ConsoleBus ERROR`		CODE EXAMPLE 4-22
`ConsolePortError` AR AR/SBBC SDC		 CODE EXAMPLE 4-5 CODE EXAMPLE 4-6 CODE EXAMPLE 4-8
L2CheckError AR AR AR SDC SDC		 CODE EXAMPLE 4-14 CODE EXAMPLE 4-15 CODE EXAMPLE 4-16 CODE EXAMPLE 4-10 CODE EXAMPLE 4-20
SafariPortError AR-AdrPErr SDC SDC SDC		 CODE EXAMPLE 4-9 CODE EXAMPLE 4-22 CODE EXAMPLE 4-6 CODE EXAMPLE 4-10 CODE EXAMPLE 4-18 CODE EXAMPLE 4-19 CODE EXAMPLE 4-22 CODE EXAMPLE 4-31
`TransactionCountOVFE rror/Underflow Errors`		CODE EXAMPLE 4-10

TABLE 1-1 Mapping of Key Error Message Identifiers To Code Examples *(Continued) (Continued)*

Identifier	Description	Code example
General Error		
PTimeout		CODE EXAMPLE 4-20
DX		CODE EXAMPLE 4-22
		CODE EXAMPLE 4-28
ErrorStatus		
SBBC-BB_ILL	Boot bus illegal access reported by SBBC	CODE EXAMPLE 4-30
SBBC-BBErr	Boot bus error reported by SBBC	CODE EXAMPLE 4-31
SBBC-BB_ILL	Boot bus illegal access reported by SBBC	CODE EXAMPLE 4-32
SBBC-PCIErrorStatus		
-SILL	Slave port detected an illegal access	CODE EXAMPLE 4-33
SBBC-DcdsEr	DCDS asserted an error	CODE EXAMPLE 4-34
SBBC-SafErr	Device asserted an error	CODE EXAMPLE 2-20
Cheetah1ErrorStatus		
DCDS-M2CBypass		CODE EXAMPLE 4-34

For the error examples given, the goal is to identify the FRU(s) responsible for the error. The action taken to resolve a given error can depend on many factors including best practices, cost of components, ease of servicing, availability of FRUs, attitude and/or risk sensitivity of the customer, etc. In some cases, there may be no action taken or, an election may be taken to upgrade the firmware. Also, you may be confronted with a list of potential FRUs and must then use a variety of methods to further isolate to the faulty FRU(s).

When possible, examples will be followed by Summary, Analysis, and Suspect FRU(s) sections to further isolate to a faulty FRU(s). Once the FRU(s) involved are identified, it will be easier determine a course of action, or to explain the problem to the customer. Note that no distinction is made between errors that occur during POST or Solaris. If both examples are available, they will be discussed in the same section.

The examples herein may be outdated in that they may be the result of firmware or hardware bugs which have been fixed. However, they should still be valid as we still need to be able to understand what each error message means. Of course, changes in firmware will change the formatting and reporting of the errors, unless the hardware changes, the errors can still occur. In summary, this document is not intended to determine POLICY, it is just to help interpret error messages and to identify failing FRUs.

Comments and suggestions are always welcome:

serengeti-support@sun

1.4 Error Logs

When troubleshooting, you need to be aware of what is and isn't presented to you. For example, debugging a domain based only on the `/var/adm/messages` output (from Explorer) may be misleading since SC error messages relating to a domain get logged to the domain console only, and are not reported to Solaris (`/var/adm/messages`). Error log sources are detailed as follows:

- `/var/adm/messages` file:

  Contains messages reported by Solaris (as determined by syslog.conf).

  No L1DX errors or any other SC messages will be printed here.

- Domain console output:

  Contains messages written to the console by Solaris (like the preceding) and SC error messages, intermixed and in the order in which they are printed by Solaris and the SC.

- `showlogs` command output

  Contains SC messages for the domain

1.5 ECC Error Messages

Solaris and the SC have different views of the system regarding error messaging, in particular, ECC events. Error messages for ECC error events detected by Solaris are mapped to the locations in memory (DIMM) that orignally contained the affected data, whereas the SC has a view of the address, control and data transfer paths. Through the use of ASIC error registers, the SC collects ECC event data that details transaction types, direction of data, sourcing or destination CPU or Schizo, and signaling syndromes which are used to flag known bad data.

For the reasons detailed in the previous paragraph, it is sometimes difficult to diagnose whether an ECC event orignated in memory or the data path. For more clarification on this issue, refer to Section C.2.4, “Determining Origin of ECC Errors” on page C-14 of the *Sun Fire 6800/4810/4800/3800 Systems Troubleshooting Manual*.

Note – For domain console errors, Solaris and SC messages will be intermixed in no particular order. Thus the following sections detail messaging by both sources so a distinction can be made between the two when they are intermixed.

1.5.1 Solaris ECC Error Messaging

ECC Error event data will be logged in a processor's asynchronous fault status register (AFSR) and the faulting physical address will be logged in the asynchronous fault address register (AFAR). The processor will then take a trap so that the error information can be logged.

Messages orignating from Solaris contain the following data:

- Time stamps, for example Radiance cases, `/var/adm/messages` file, or errID info from Solaris message, see CODE EXAMPLE 1-1
- AFT (asynchronous fault tag) number

  The different AFT tag values are:

  - AFT0 - correctable errors
  - AFT1 - uncorrectable errors as well as for errors that result in a panic
  - AFT2 and AFT3 - logging diagnostic data and other error related messaging

- AFSRs (asynchronous fault status register) and AFARs (asynchronous fault address register) outputs

CODE EXAMPLE 1-1 Solaris Error Message

```
[AFT0]Corrected system bus (CE) Event on CPU18 at TL=0, errID
0x0000c9b9.19d92690
AFSR 0x00000002<CE>.00000097 AFAR 0x00000001.04bdf7d0
Fault_PC 0x10024a74 Esynd 0x0097 /N0/SB5/P3/B0/D2 J16500
[AFT0] errID0x0000c9b9.19d92690 Corrected Memory Error on /N0/SB5/P3/B0/D2
J16500 is Persistent
[AFT0] errID0x0000c9b9.19d92690 Data Bit 3 was in error and corrected
```

1.5.2 SC ECC Error Messaging

- Time stamped
- Reporting ASIC type/number (for example DX1)
- “From” and “For” error messages (source and destination)

CODE EXAMPLE 1-2 SC Error Message

```
Oct 25 09:00:31 wpc26sc0 Domain-A.SC [ID 542264 local1.error] /N0/SB4 reported ECC error
Oct 25 09:00:32 wpc26sc0 Domain-A.SC [ID 804061 local1.error] Status: 0x81210000
Syndrome from DX2: 0x00000000
Syndrome from DX3: 0x80970000
Read transaction for Cheetah C: DTrans: 0x121 (AID=18, SEQ#=1)
         ECC Syndrome: 0x097
         MTag Syndrome: 0x0
Oct 25 09:00:32 wpc26sc0 Domain-A.SC [ID 542266 local1.error] /N0/SB5 reported ECC error
Oct 25 09:00:32 wpc26sc0 Domain-A.SC [ID 506171 local1.error] Status:  0xe1210000
Syndrome from DX2: 0x00000000
Syndrome from DX3: 0x00970000
Read transaction from Cheetah D: DTrans: 0x121 (AID=18, SEQ#=1)
         ECC Syndrome: 0x097
```

CHAPTER 2

CPU / Schizo and CPU / Schizo Related Errors

2.1 Overview

The following sections detail errors encountered which may be attributed to either a CPU, it's Ecache or a Schizo and, where applicable, how these errors may be propagated onto the data path.

2.2 CPU ECC Error Messaging

When an ECC error event is detected, event data will be logged in a CPU's asynchronous fault status register (AFSR) and the faulting physical address will be logged in the asynchronous fault address register (AFAR). The CPU will then take a trap so that the error information can be logged. For example:

Each message is tagged with an asynchronous fault tag (AFT) to identify the data being logged. Continuation messages begin with four spaces. The different AFT tag values are

- AFT0 is used for correctable errors
- AFT1 is used for uncorrectable errors as well as for errors that result in a panic

- AFT2 and AFT3 are used for logging diagnostic data and other error related messaging

CODE EXAMPLE 2-1 Sample Solaris Error Message

```
[AFT0]Corrected system bus (CE) Event on CPU22 at TL=0, errID
0x0000c9b9.19d92690

AFSR 0x00000002<CE>.00000097 AFAR 0x00000001.04bdf7d0

Fault_PC 0x10024a74 Esynd 0x0097 /N0/SB5/P3/B0/D2 J16500

[AFT0] errID0x0000c9b9.19d92690 Corrected Memory Error on /N0/SB5/P3/B0/D2
J16500 is Persistent

[AFT0] errID0x0000c9b9.19d92690 Data Bit 3 was in error and corrected
```

- errID is a timestamp of the event. This is very useful for correlating multiple errors that occurred at the same time.
- AFSR and AFAR are the asynchronous fault status and address registers.
- Fault_PC is the value of the PC at the time of the fault and is dependent upon the fault type as to whether the value is valid.
- Esynd is the ECC syndrome captured
- /N0/SB5/P3/B0/D2 is the identifier of the memory module which corresponds to the faulting address
- J16500 is the J number for that memory module

Note – The definitions of intermittant, persistant and sticky may change due to Bug ID # 4497348.

- Solaris describes this event as Persistent. The Solaris software error handling code provides a disposition code as a result of the scrub operation. This disposition is one of Intermittent, Persistent, or Sticky. The definition of each of these codes is:
  - Intermittent means the error was not detected on a reread of the affected memory location.
  - Persistent means the error was detected again on a reread of the affected memory location but the scrub operation corrected it. See Section , “Errors Categorized as Persistent” on page C-15, of the *Sun fire 6800.*
  - */4810/4800/3800 Systems Troubleshooting Manual* for servicing information.

- Sticky means that the error still exists in memory even after the scrub operation. These events should be investigated further to determine if some hardware replacement is necessary since this is indicative of a hard failure. See Section , “Errors Categorized as Sticky” on page C-16 for servicing information.

The error in CODE EXAMPLE 2-1can be categorized as a correctable memory error on memory module /N0/SB5/P3/B0/D2. The scrub operation successfully cleared the error in memory.

Here are some general notes for Solaris Ecache ECC messages.

- The most important information is the CPU number. This will tell you the FRU affected by the error, see TABLE 2-1.
- User or privileged mode or trap level (TL) are irrelevant from the standpoint of isolating the failure.
- When multiple bits are present in the AFSR, the Solaris handler prints a block of messages for each error bit. For multiple errors, it is not clear which error is associated with the captured physical address stored in the AFAR, therefore Solaris decodes the AFSR error priority. This results in the decoding described in TABLE 2-2.

TABLE 2-1 CPU / Schizo Agent IDs

SB	CPU A (0)	CPU B (1)	CPU C (2)	CPU D (3)
SB0	0 (0x00)	1 (0x01)	2 (0x02)	3 (0x03)
SB1	4 (0x04)	5 (0x05)	6 (0x06)	7 (0x07)
SB2	8 (0x08)	9 (0x09)	10 (0x0a)	11 (0x0b)
SB3	12 (0x0c)	13 (0x0d)	14(0x0e)	15(0x0f)
SB4	16 (0x10)	17 (0x11)	18 (0x12)	19 (0x13)
SB5	20 (0x14)	21 (0x15)	22 (0x16)	23 (0x17)
IB	**Schizo 0**	**Schizo 1**		
IB6	24 (0x18)	25 (0x19)		
IB7	26 (0x1a)	27 (0x1b)		
IB8	28 (0x1c)	29 (0xd)		
IB9	30 (0x1e)	31 (0x1f)		

TABLE 2-2 Solaris AFAR Validity

Decoded value	Description
Ambiguous	AFAR may or may not be valid for this error bit (two or more errors occurred with equal priority)
Invalid	AFAR is not valid for this error bit (one or more errors have high priority)
Nothing	AFAR is likely valid (this error has the highest priority)

Note – Be advised that in all cases, Solaris decodes the AFAR to the system board and Ecache DIMM (even in cases when it is not sure that AFAR is valid). See cheetah.c for more information

2.2.1 CPU Ecache Errors

When a CPU detects an ECC error in it's Ecache, it will take a trap and report one or more error bits in it's AFSR register. These bits (for cheetah and cheetah+) are detailed in TABLE 2-3 and illustrated in FIGURE 2-1 as follows:

TABLE 2-3 CPU Ecache Error Bits

Error bits	Definitions
UCC,UCU,EDC,EDU	ECC errors found while CPU is accessing it's own Ecache
EDU:BLD	Uncorrectable Ecache error for block load
WDC, WDU	ECC errors found while CPU is pushing data out of it's Ecache
CPC, CPU	ECC errors found while CPU is supplying data out of it's Ecache to another CPU
EMC,EMU	Mtag errors incoming to a CPU

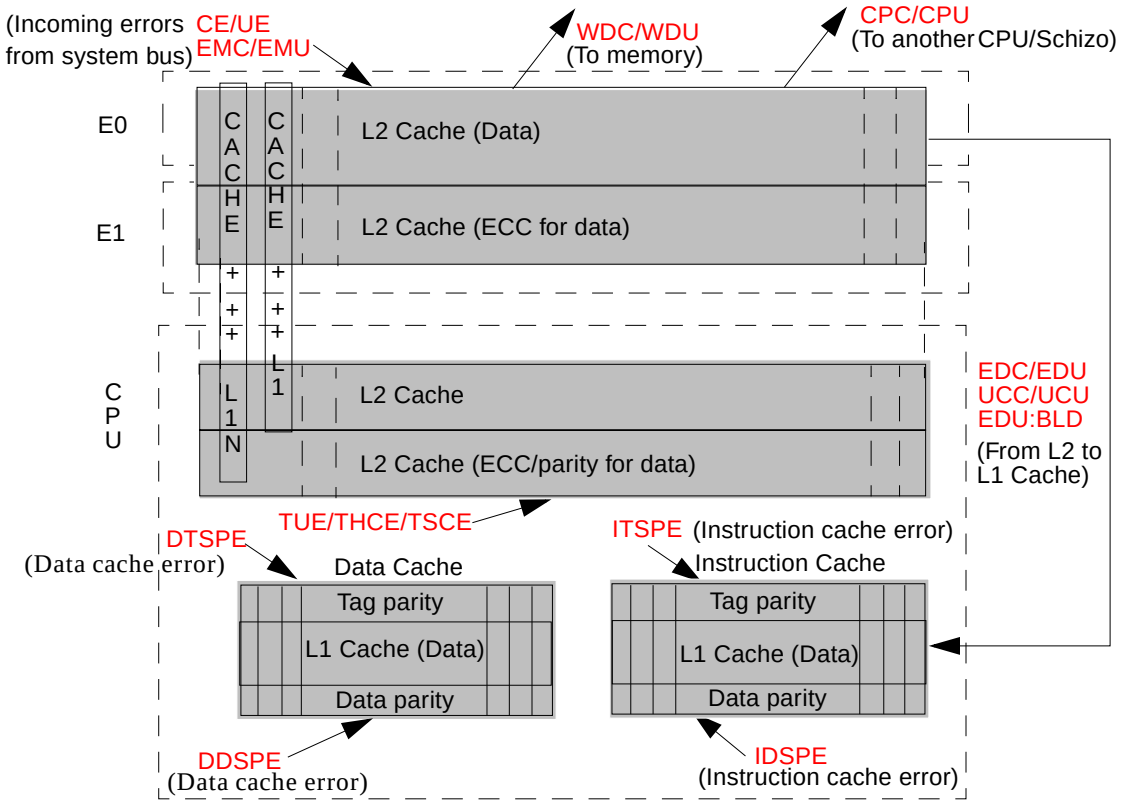


FIGURE 2-1 UltraSPARC III+ Error Diagram

Note the following:

- See TABLE 2-3 and TABLE 2-6 for error definitions
- Correctable errors are not propagated (e.g. WDC, CPC, EDC, UCC)
- Note that L2 cache information is contained in both the UltraSPARC III+ chip and DIMMS (see dotted lines connecting both boxes)
- Some of these errors can only occur in UltraSPARC III+
- Correctable errors end in C, uncorrectable errors end in U
- Solaris may or may not be able to recover from an instance of these errors.
- Multiple errors (of the same or different types) can occur at the same time.

The only differences between these error types depends upon what the processor is doing when the error is first detected. Thus, from a troubleshooting standpoint, these errors are all the same with the affected FRU being the reported system board.

When a CPU sends known bad data out of it's Ecache into memory or, to another CPU or Schizo, then the CPU will intentionally force bad ECC with a special signaling syndrome on the data as it is sent out. Some of these errors will result in further errors being reported by the ASICs in the data path (such as L1DX ECC errors). Thus, a CPU Ecache ECC error can, and often will, turn into a system data path ECC error.

Note that the syndrome associated with bad data is called a signalling syndrome and can help distinguish these cases from system ECC errors, see legend for FIGURE 2-1 and the following section for more signalling syndrome information.

2.2.1.1 Interpreting Syndromes

Use TABLE 2-4 to find the bit or bits in error using the ECC syndrome ouput with ECC error messsages, see Chapter 3 and Section 3.3.3.1, "How to Use Syndrome Table" on page 3-12

TABLE 2-4 ECC Syndrome UltraSPARC III+ and Schizo (See legend that follows Table)

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	----	C0	C1	M2	C2	M2	M3	47	C3	M2	M2	53	M2	41	29	M
1	C4	M	M	50	M2	38	25	M2	M	33	24	M2	11	M	M2	16
2	C5	M	M	46	M2	37	19	M2	M	31	32	M	7	M2	M2	10
3	M2	40	13	M2	59	M	M2	66	M	M2	M2	0	M2	67	71	M
4	C5	M	M	43	M	36	18	M	M2	49	15	M	63	M2	M2	6
5	M2	44	28	M2	M	M2	M2	52	68	M2	M2	62	M2	M3	M3	M4
6	M2	26	106	M2	64	M2	M2	2	120	M	M2	M3	M	M3	M3	M4
7	116	M2	M2	M3	M2	M3	M	M4	M2	58	54	M2	M	M4	M4	M3
8	C7	M2	M	42	M	35	17	M2	M	45	14	M2	21	M2	M2	5
9	M	27	M	M2	99	M	M	3	114	M2	M2	20	M2	M3	M3	M
A	M2	23	113	M2	112	M2	M	51	95	M	M2	M3	M2	M3	M3	M2
B	103	M	M2	M3	M2	M3	M3	M4	M2	48	M2	M	73	M2	M	M3
C	M2	22	110	M2	109	M2	M	9	108	M2	M	M3	M2	M3	M3	M
D	102	M2	M	M	M2	M3	M3	M	M2	M3	M3	M2	M	M4	M	M3
E	98	M	M2	M3	M2	M	M3	M4	M2	M3	M3	M4	M3	M	M	M
F	M2	M3	M3	M	M3	M	M	M	56	M4	M	M3	M4	M	M	M
10	C8	M	M2	39	M	34	105	M2	M	30	104	M	101	M	M	4
11	M	M	100	M	83	M	M2	12	87	M	M	57	M2	M	M3	M

TABLE 2-4 ECC Syndrome UltraSPARC III+ and Schizo (See legend that follows Table) *(Continued)*

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
12	M2	97	82	M2	78	M2	M2	1	96	M	M	M	M	M	M3	M2
13	94	M	M2	M3	M2	M	M3	M	M2	M	79	M	69	M	M4	M
14	M2	93	92	M	91	M	M2	8	90	M2	M2	M	M	M	M	M4
15	89	M	M2	M3	M2	M	M3	M	M2	M	M3	M	M3	M	M	M3
16	86	M	M2	M3	M2	M	M3	M	M2	M	M3	M	M3	M	M	M3
17	M	M	M3	M2	M3	M2	M4	M	60	M	M2	M3	M4	M	M	M2
18	M2	88	85	M2	84	M	M2	55	81	M2	M2	M3	M2	M3	M3	M4
19	77	M	M	M	M2	M3	M	M	M2	M3	M3	M4	M3	M2	M	M
1a	74	M	M3	M3	M	M	M3	M	M	M	M3	M	M3	M	M4	M3
1b	M2	70	107	M4	65	M2	M2	M	127	M	M	M	M2	M3	M3	M
1c	80	M2	M2	72	M	119	118	M	M2	126	76	M	125	M	M4	M3
1d	M2	115	124	M	75	M	M	M3	61	M	M4	M	M4	M	M	M
1e	M	123	122	M4	121	M4	M	M3	117	M2	M2	M3	M4	M3	M	M
1f	111	M	M	M	M4	M3	M3	M	M	M	M3	M	M3	M2	M	M

The following legend applies to TABLE 2-4.

Error bit	Description
----	No error
0 to 127	Single data bit error
C0 to C8	Single bit error on check bits 0 to 8
M2	Probable double bit error within a nibble
M3	Probable triple bit error within a nibble
M4	Probable quad bit error within a nibble
M	Multibit error
0x003, 0x71, 0x11c	Special ECC Signaling Syndromes (see following section)

2.2.1.2 Special ECC Signalling Syndromes

Some ECC syndromes are reserved by hardware to indicate an intentional error. The hardware does this to flag data that it knows is bad, but needs to pass on anyway. The signalling ECC syndromes (which map to uncorrectable errors) tells the end-user of the data that it is no good. The actual value of the data generated depends upon the situation.

There are three signalling ECC values which can occur in four situations, TABLE 2-5

TABLE 2-5 Special ECC Syndromes

Syndrome	Situation
0x003	Generated by a CPU as the result of store-merge operation to a line that already has an uncorrectable ECC in it's Ecache.
0x003	Generated by a Schizo as the result of a partial-line DMA writes to a line that was previously read from the system bus with UE ECC
0x071	Generated by a CPU as the result of a writeback or copyout of a line that previously contained UE ECC
0x11c	Generated by a CPU as the result of a Bus Error

If you see these syndromes in any ECC error (reported by a CPU, Schizo, or L1DX) you need to check and see if this error was caused by some prior error event. Since these syndromes map to uncorrectable errors (specifically, certain double-bit errors), it is possible (but unlikely) that these could be the result of real errors.

Specifically, 0x071 is the syndrome you will see when bad data is written from a CPU's Ecache to another CPU or Schizo (as the result of a copyout or writeback). When a Schizo has to write bad data (to another CPU) as the result of a partial-line DMA write on a line that it already received with bad ECC, it will use the syndrome of 0x003.

Note that in either case (CPU or Schizo), the original bad ECC present in the cache is not used! As such, the only signalling ECC syndrome values that will be seen by an L1DX are 0x071 (data originated from a CPU) and 0x003 (data originated from a Schizo). These are the most important signalling ECC values and you will need to look further to find the original source of the error.

0x11c is the syndrome you will see when bogus data is inserted into the Ecache as a placeholder after a Bus Error (BERR) occurs on a cacheable memory read. The CPU will see the BERR first, then later may encounter an Ecache ECC error with a syndrome of 0x11c if it later tries to access the same Ecache location. Therefore, if you see a BERR followed by any CPU Ecache ECC error with syndrome 0x11c, you can ignore the ECC errors.

Note that a cacheable load resulting in a system timeout error (TO) will insert garbage data into the Ecache and any result is possible. Of course, the system will see the TO trap first (as is the case for BERR).

For an incoming UE error on a cacheable load, the Ecache will retain it's same bad syndrome value as was present on the system bus.

Finally, 0x003 is the syndrome you will see also when the processor encounters bad ECC in it's external cache as it is writing data into it (called a store-merge). This will first result in an EDU trap being taken. Afterward the store merge completes, the external cache line which had bad ECC will be forced to have the syndrome of 0x3.

FIGURE 2-2 illustrates the states that a CPU or Schizo cache line may transition through for various error cases.

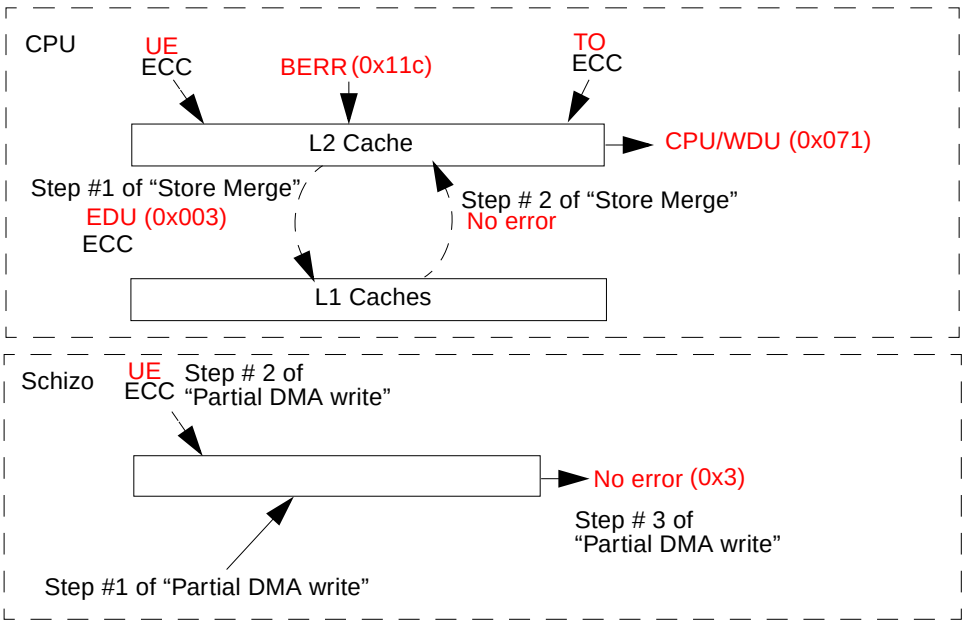


FIGURE 2-2 Syndrome States for a Cache Line

Note: On a UE, UCU, EDU, I'm not sure what parity value gets inserted into the L1 Caches on USIII+ (assuming the cache line is not invalidated).

Syndrome Behavior on BERR, TO, and UE

I have no examples of store-merge errors with syndrome 0x003, or any ECC errors after BERR or TO traps. PLEASE send me examples if you have them.

To put this all in perspective, the following (highly unlikely!) scenario is possible:

1. **A BERR event occurs, causing a cache line to have bad ECC (syndrome 0x11c).**
2. **A subsequent store-merge to the same line would force the bad ECC to change (syndrome 0x003). This would cause an EDU event.**
3. **A subsequent write-back of the same line would force the bad ECC to change again (syndrome 0x071) as it went out on the system bus. This would be then seen as an WDU or CPU event by the CPU as well as zero or more L1DX ECC errors by the SC.**

2.2.1.3 ECC errors on Interrupt Vectors

When a CPU or Schizo needs to send an interrupt to another CPU, it sends a special bus packet. This bus packet is protected by ECC just like any other data transaction on the Safari Bus. If an ECC error is noticed by the receiving CPU, it will take a trap and record one of the following bits in it's AFSR.

- IVC-H/W corrected system bus data ECC error for read of interrupt vector data
- IVU-Uncorrectable system bus data ECC error for read of interrupt vector data

You can treat these just like a CE/UE error except that you know the source wasn't a memory DIMM, it was a CPU or Schizo (but you don't know which one). You will need to correlate this to any L1DX ECC errors reported to locate the source. Fortunately, the syndrome is captured, so this might help you match it to an L1DX error.

2.2.2 Ecache Tag ECC and Data/Instruction Cache Tag Parity Errors

The errors listed in TABLE 2-6 and illustrated in FIGURE 2-1 can only occur with Cheetah +.

TABLE 2-6 Ecache Tag ECC and Data/Instruction Cache Tag Parity Errors

AFSR bit	Description	Code example
TSCE	Software corrected, single-bit External Cache tag ECC error	CODE EXAMPLE 2-2
THCE	Hardware corrected, single-bit External Cache tag ECC error	CODE EXAMPLE 2-3
DDSPE	Primary Data Cache data or tag parity error	CODE EXAMPLE 2-4
DTSPE	Primary Data Cache tag parity error	CODE EXAMPLE 2-5
IDSPE	Primary Instruction Cache data parity error	CODE EXAMPLE 2-6
ITSPE	Primary Instruction Cache tag parity error	CODE EXAMPLE 2-7
TUE	Uncorrectable, multi-bit External Cache tag ECC error	

In the case of a TUE error, the CPU will assert it's ERROR pin, so a SYSTEM ERROR is generated which will be covered later in this text. No panic or recovery is possible. UE and CE errors are the result of external errors seen by the CPU. Thus, you need to look for L1 DX ECC errors to see if you can locate the source.

For all remaining errors (including TUE), they indicate problems with the reporting CPU. These errors may or may not be recoverable by Solaris. The FRU, if any were to be swapped, would be the containing system board.

The following sections contain examples of each of the errors listed in TABLE 2-6.

2.2.2.1 Example of Software Correctable ECC Ecache Tag Error (TSCE)

CODE EXAMPLE 2-2 Software Correctable ECC Ecache Tag Error

```
SUNW,UltraSPARC-III+: NOTICE: [AFT0] TSCE Event on CPU14 in Privileged mode at
TL=0, errID 0x0000013c.f98a34f0   AFSR 0x01100000<TSCE,PRIV>.00000000 AFAR
0x00000000.3f3b0000  Fault_PC 0x30001aab084
SUNW,UltraSPARC-III+: [AFT2] errID 0x0000013c.f98a34f0 PA=0x00000000.3f3b0000
E$tag 0x00000000.fc000004 E$state_0 Modified
SUNW,UltraSPARC-III+: [AFT2] E$Data (0x00) 0x0eccf00d.ff270000
0x0eccf00d.ff270000 ECC 0x1fb
SUNW,UltraSPARC-III+: [AFT2] E$Data (0x10) 0x0eccf00d.ff270000
0x0eccf00d.ff270000 ECC 0x1fb
SUNW,UltraSPARC-III+: [AFT2] E$Data (0x20) 0x0eccf00d.ff270000
0x0eccf00d.ff270000 ECC 0x1fb
SUNW,UltraSPARC-III+: [AFT2] E$Data (0x30) 0x0eccf00d.ff270000
0x0eccf00d.ff270000 ECC 0x1fb
SUNW,UltraSPARC-III+: [AFT2] D$ data not available
SUNW,UltraSPARC-III+: [AFT2] I$ data not available
```

Analysis

This is a single bit, software correctable Ecache tag ECC error (TSCE, see TABLE 2-6). The responsible FRU would be SB3 which contains CPU 14.

Suspect FRU

- SB3

2.2.2.2 Example of Hardware Correctable ECC ECache Tag Error (THCE)

CODE EXAMPLE 2-3 Hardware Correctable ECC Ecache Tag Error

```
SUNW,UltraSPARC-III+: NOTICE: [AFT0] THCE Event on CPU14 at TL=0, errID
0x00000165.86cacb90  AFSR 0x02000000<THCE>.00000000 AFAR 0x00000000.23348000
Fault_PC 0x30001aab410
SUNW,UltraSPARC-III+: [AFT2] errID 0x00000165.86cacb90 PA=0x00000000.23348000
E$tag 0x00000000.8c000004 E$state_0 Modified
SUNW,UltraSPARC-III+: [AFT2] E$Data (0x00) 0x0eccf00d.ff270000
0x0eccf00d.ff270000 ECC 0x1fb
SUNW,UltraSPARC-III+: [AFT2] E$Data (0x10) 0x0eccf00d.ff270000
0x0eccf00d.ff270000 ECC 0x1fb
SUNW,UltraSPARC-III+: [AFT2] E$Data (0x20) 0x0eccf00d.ff270000
0x0eccf00d.ff270000 ECC 0x1fb
SUNW,UltraSPARC-III+: [AFT2] E$Data (0x30) 0x0eccf00d.ff270000
0x0eccf00d.ff270000 ECC 0x1fb
SUNW,UltraSPARC-III+: [AFT2] D$ data not available
SUNW,UltraSPARC-III+: [AFT2] I$ data not available
```

Analysis

This is a hardware correctable Ecache tag ECC errors (THCE, see TABLE 2-6). The responsible FRU would be SB3 which contains CPU 14.

Suspect FRUs

- SB3

2.2.2.3 Example of Data Cache Parity Errors from a CPU (DDSPE)

CODE EXAMPLE 2-4 Data Cache Parity Error from a CPU

```
NOTICE: [AFT0] DDSPE Event on CPU14 in Privileged mode at TL>0, errID
0x0000007d.61ffb610    Fault_PC 0x1032cf4
[AFT2] D$Parity (0x0:0:0x00) 0x00
[AFT2] D$Data (0x00) 0x0eccf00d.ff270001 *Bad* 0x0eccf00d.ff270000
[AFT2] D$Data (0x10) 0x0eccf00d.ff270000 0x0eccf00d.ff270000
[AFT2] D$Tag (0x0:0) 0x40035a7d D$state Valid D$utag 0x007f D$snp 0x40035a7c
[AFT2] PAtag 0x000.35a7c000 PAsnp 0x000.35a7c000 VAutag 0x1fc000

panic[cpu14]/thread=300020912a0: [AFT1] errID 0x0000007d.61ffb610 DPE Error(s)
 See previous message(s) for details
```

Analysis

Both of these are Dcache data parity errors. One occurred in kernel mode (TL>0). The responsible FRU would be SB3 which contains CPU 14 or 15.

Suspect FRU

- SB3

2.2.2.4 Example of Data Cache Tag Parity Error from a CPU (DTSPE)

CODE EXAMPLE 2-5 Dcache Tag Parity Error from a CPU

```
SUNW,UltraSPARC-III+: NOTICE: [AFT0] DTSPE Event on CPU14 in Privileged mode at
TL=0, errID 0x000000c9.e1a35e20    Fault_PC 0x30001aab084
SUNW,UltraSPARC-III+: [AFT2] D$Tag (0x0:1) 0x4003d35f *Bad* D$state Valid D$utag
0x00aa D$snp 0x4003d35c
SUNW,UltraSPARC-III+: [AFT2] PAtag 0x000.3d360000 *Bad* PAsnp 0x000.3d35e000
VAutag 0x2aa000
```

Analysis

This is a Dcache tag parity error. The responsible FRU would SB3 which contains CPU 14.

Suspect FRU

- SB3

2.2.2.5 Example of Instruction Cache Data Parity Error from a CPU (IDSPE)

CODE EXAMPLE 2-6 InstructionCache Tag Parity Error from a CPU

```
UNW,UltraSPARC-III+: NOTICE: [AFT0] IDSPE Event on CPU14 in Privileged mode at
TL=0, errID 0x000000c5.54f189b0
     Fault_PC 0x30001aab000
SUNW,UltraSPARC-III+: [AFT2] I$Data (0x80:0:0x00) 0x003.92100001 *Bad*
SUNW,UltraSPARC-III+: [AFT2] I$Tag (0x80:0) 0x00.01fa2600 I$state Valid I$utag
0x55 I$vpred 0x0.0 I$snp 0x00.01fa2600
SUNW,UltraSPARC-III+: [AFT2] PAtag 0x000.3f44d000 PAsnp 0x000.3f44d000 VAutag
0x0ab000
```

Analysis

This is an Icache data parity error. It occurred in kernel mode (TL>0). The responsible FRU would be SB3 which contains CPU 14.

Suspect FRU

- SB3

2.2.2.6 Example of Instruction Cache Data Parity Error from a CPU (ITSPE)

The following example was from a Sun Fire 6800 system.

CODE EXAMPLE 2-7 Instruction Cache Tag Parity Error from a CPU

```
SUNW,UltraSPARC-III+: NOTICE: [AFT0] ITSPE Event on CPU15 in Privileged mode at
TL=0, errID 0x000000c7.73fba7d0    Fault_PC 0x30001aab000
SUNW,UltraSPARC-III+: [AFT2] I$Tag (0x80:0) 0x20.01996b00 *Bad* I$state Valid
I$utag 0x55 I$vpred 0x0.0 I$snp 0x20.01996a00
SUNW,UltraSPARC-III+: [AFT2] PAtag 0x000.332d7000 *Bad
```

Analysis

This is an Icache tag parity error. It occurred in kernel mode (TL>0). The responsible FRU would SB3 which contains CPU 15.

Suspect FRU

- SB3

2.2.3 MTag Errors

The Mtag is a 3 bit wide field used to implement SSM (wildcat) mode. However, the CPU (and presumably L1DX's) will check Mtag ECC correctness regardless of the mode (SSM or not). Likewise, EMC and EMU traps are enabled (in fact, can't be disabled) regardless of mode. Mtags are protected by ECC on the system bus. The syndrome that reports correctable and uncorrectable errors in the Mtag is 4 bits wide.

There are no signalling values for the Mtag Syndrome. Single-bit Mtag Syndrome values are: 0x7, 0xb, 0xd, 0x1, 0x2, 0x4, and 0x8. Multi-bit syndromes are everything else, except 0 (which indicates no error).

See tables P-8, P-9 in the JPS1 Implementation Supplement for more information.

NOTE: What about Schizo? Does it check mtags?

TABLE 2-7 ECC Mtag Errors

AFSR bit	Description
EMC	Hardware corrected system bus Mtag ECC error
EMU	Uncorrectable, multi-bit system bus Mtag ECC error

In the case of EMU and TUE errors, the CPU will assert it's ERROR pin, so a SYSTEM ERROR is generated which will be covered later in this text. No panic or recovery is possible.

In the case of EMC and EMU errors, these can be treated in much the same way as UE and CE errors. In other words they are the result of external errors seen by the CPU. Thus, you need to look for L1 DX Mtag ECC errors to see if you can locate the source. However, unlike UE and CE errors, Mtag errors are never the result of DRAM failures.

All remaining errors, including TUE, indicate problems with the reporting CPU. These errors may or may not be recoverable by Solaris. The FRU, if any were to be swapped, would be the containing system board.

2.2.4 CPU Timeout (TO / DTO) and Bus Errors (BE / DBERR)

CPU bus and timeout errors result while accessing a physical address. These errors are reported through assertion of the AFSR bits listed in table TABLE 2-8.

TABLE 2-8 ASFR CPU Timeout and Bus Error Bits

ASFR bit	Definition
TO, DTO	Timout errors
BERR, DBERR	Bus errors

A timeout error occurs when no device responds with a MAPPED status as the result of the system bus address phase. This means that a CPU (or schizo?) tried to access an address range not claimed as owned by any other system board.

A Bus Error occurs when an error is returned on a system bus read operation. This means a CPU (or schizo?) tried to read an address (which was mapped), yet an error was returned for an unknown reason.

Bus or timeout errors can be caused by a system software bug or bad hardware. Most important, for either error type, is the CPU reporting the error and the AFAR (which represents the device being addressed). Generally, the fault lies in either of these two devices or the system software. In addition, the core file can be useful as well.

2.2.4.1 Example of Timeout Error (TO)

The following example is from a single segment Sun Fire 6800 system with Domain A containing SB0, SB3 and SB4. The source is from America Radiance Case #62653540.

CODE EXAMPLE 2-8 Timeout Error

```
CPU13 Privileged Timeout Error: AFSR 0x00101000.00000000 AFAR
0x00000300.4fcfc720
```

Analysis

This is a timeout error that occurred on CPU 13 while it was in privileged mode (i.e. In the kernel). The fact that privileged is noted is not really important.

First decode the AFAR which will indicate what the CPU was trying to talk to when the error occurred. You can use one of several references to help you with this:

- Serengeti Architecture Programmer's Reference, chapter 2
- SRDB #28506 (probably the easiest...)

In this case, addresses that are less than 0x400.0000.0000 always refer to main system memory (i.e. another CPU/memory board). Which board, is a good question.

We do know that COU13 (SB3, CPU B) was the one who noticed the problem. The other suspect FRUs include any other system board.

Case note: SB3 was swapped and the incident was closed.

Hint to find TO errors in Sunsolve, search for AFSR 0x00101000.00000000 or 0x00001000.00000000 (Privileged or non-privileged timeout errors).

2.2.4.2 Example of `Privileged Bus Error Panic`

The following example is from a single segment Sun Fire 4800 system with Domain A containing SB0 and SB4. The source is America Radiance case #62445499. The system panicked twice with the same message.

CODE EXAMPLE 2-9 Privileged Bus Error

```
CPU2 Privileged Bus Error: AFSR 0x00100800.00000000 AFAR 0x00000638.09808000
```

Analysis

This is a problem between CPU #2 (SB0, CPU C) and some other device. As before, we need to decode the AFAR. This time we have an address that points to a non-cacheable address and likely an I/O board.

Looking at the bits above 31, we have (0x638). This breaks down as follows

0x638 == 0b0110.0011.1000

Dropping the first bit #43 (a 0, but it isn't valid), we get:

0b110.0011.1000

The first 1 bit means that this is an I/O address. The second 1 bit means it is in the device pair bus space.

The next three bits is the node ID (000). These bits are always the same for (non-Wildcat) Sun FIre.

The next five bits are the agent ID (11100) or 0x1c. This maps to IB8, Schizo #0.

WARNING: The last bit (a 0 in this case) indicates PCI bus B, a 1 bit would indicate PCI bus A (counter-intuitive).

With the last two pieces of information (IB8, Schizo #0 and PCI bus B), and the fact that we know this is an 4800, we can attempt to map this to a actual physical PCI slot.

Referring to SRDB 28506, on a 4800, we can only have an 8 slot PCI or 4 slot cPCI in IB8. So, depending upon the actual configuration:

If an eight slot PCI: AID 0x1c PCI leaf B maps to IB 8, slot 0,1 or 2

If a four slot cPCI: the same info maps to IB 8, slot 2

NOTE: In this case we have an eight slot PCI so the slot number is ambiguous.

From the core dump that is generated for BERR and TO cases, it is often helpful to look at the stack trace. In this case we are using the field tool FM:

CODE EXAMPLE 2-10 Core Dump

```
unix:panicsys+0x44 (0x10456ce0, 0x2a1001fd7b0, 0x101416e8, 0x78002000,
0x30005558000, 0xf)
unix:vpanic+0xcc (0x101416e8, 0x2a1001fd7b0, 0x1, 0x213cd250, 0x213cd252, 0x0)
genunix:cmn_err+0x28 (0x3, 0x101416e8, 0x2, 0x104cd9c8, 0x104cd9d0, 0x104cd9f0)
SUNW,UltraSPARC-III:cpu_async_error+0x630 (0x10080000000000, 0x63809808000, 0x0,
0x2a1001fd920, 0x0, 0x219)

unix:prom_rtt+0 (0x30000814f08, 0x3000006c00a, 0x20, 0xd00000000, 0xd, 0x1fff)
-- prom_rtt regs data  rp: 0x2a1001fd920
pc:  0x10032954 unix:i_ddi_get16+0x4:           lduh    [%o1], %o0
npc: 0x1020c84c isp:isp_intr+0x68:      andcc   %o0, 0x4        ( btst  %o0, 0x4 )

isp:isp_intr+0x60 (0x4400, 0x1000, 0x0, 0x0, 0x0, 0x30000a86468)
pcisch:pci_intr_wrapper+0x64 (0x104fd978, 0x700, 0x1, 0x30000965410,
0x30000820c88, 0x30000984738)
unix:intr_thread+0xa4 (0x1ef, 0x24fac9d, 0xe160bfbb, 0x9, 0x20000000, 0x0)

unix:prom_rtt+0 (0x31, 0x3000a45ae80, 0x10200, 0x2a100a65ba0, 0x4411001606, 0x1)
-- interrupt data  rp: 0x2a100a652f0
pc:  0x1001a978 unix:sfmmu_tlb_demap+0x178:             andcc   %g2, %g3
( btst  %g2 %g3 )
npc: 0x1001a97c unix:sfmmu_tlb_demap+0x17c:             be,a,pt %xcc,
unix:sfmmu_tlb_demap+0x1a0 (1f)

unix:sfmmu_tlb_demap+0x178 (0x0, 0x3000080ff18, 0x213cd220, 0x0, 0x30005558000,
0x80004)
unix:sfmmu_hblk_unload+0x1c8 (0x4, 0x213cd220, 0x4, 0x213cd250, 0x213cd252,
0x2000)
unix:hat_unload+0x3e0 (0x30005558000, 0x10, 0x1, 0x30005558000, 0x1,
0x3000080ff18)
mm:mmio+0xa4 (0x7821dcb8, 0x0, 0x2000, 0x0, 0x64f3d, 0x2a100a65a28)
mm:mmrw+0x11c (0xd00000000, 0x2a100a65a28, 0x100000, 0xd00000000, 0xd, 0x1fff)
mm:mmread+0x10 (0xd00000000, 0x2a100a65a28, 0x3000418ff28, 0x0, 0x30005f0ad30,
0xd00000000)
specfs:spec_read - frame recycled
genunix:read+0x25c (0xc9e00000, 0x0, 0x2001, 0x30005f4c9e8, 0x5, 0x100000)
unix:syscall_trap+0x88 (0x5, 0x10023d340, 0x100000, 0x0, 0x100110af8, 0x0)
```

Without knowing too much about stack traces, we can see the last fault (BERR) occurred while in the ISP driver (the stack is display in reverse time order). We could examine the core dump further to see which instance was at fault, but in our case (looking at the prtdiag output that follows), there is only one isp:

CODE EXAMPLE 2-11 prtdiag Output

```
 Bus  Max
            IO   Port Bus       Freq Bus  Dev,
FRU Name   Type ID  Side Slot MHz  Freq Func State Name
Model
----------       ----    ---- ----    ----     ----   ----     ----    -----
--------------------------------  ----------------------
...
N0/IB8/P0  PCI   28   B    0    33   33  4,0  ok    SUNW,isptwo-pci1077,1020/sd
(blo+ QLGC,ISP1040B
...
```

Note in the string N0/IB8/P0, that P0 indicates the Schizo number. So this card is in IB8, Schizo #0, PCI leaf B, slot #0. This is consistent with the AFAR that we decoded above.

Question: is there a way of mapping the remaining AFAR bits (31:0) to narrow down the slot number (or) to determine if this is a valid address for the referenced card... HMMM.... Would prtconf or the device tree be helpful?

The above analysis is extremely helpful as we now know exactly which card CPU 2 was trying to talk to (in this case an USWIS isp/hme). This narrows the suspect FRU list to (CPU 2 on SB0, IB8, or the above PCI card). Also, don't eliminate the posibility of a bad driver or software causing this problem.

At this point, without further information, it is a guess as to what the real problem is. In this case, it was recommended to blacklist CPU 2 and move the affected PCI card to another slot (on a different schizo) and wait for a third occurance.

2.2.4.3 Examples of of Panic Incidents

The source is AMER Radiance case #62667973. In this case we have 9 incidents of panics and one of a system error:

CODE EXAMPLE 2-12 Panic Incident 1

```
Sep 18 08:21:48 prs-sun-20-nyth-phitx in.mpathd[44]: [ID 299542 daemon.error]
NIC repair detected on qfe0 of group front-vlan
Sep 18 16:46:05 prs-sun-20-nyth-phitx SUNW,UltraSPARC-III: [ID 582682
kern.warning] WARNING: [AFT1] Bus Error (BERR) Event on CPU8 Privileged Data
Access at TL=0, errID 0x00001bc0.06e19b00
Sep 18 16:46:05 prs-sun-20-nyth-phitx     AFSR 0x00000800<BERR>.00000000 AFAR
0x00000630.0d806330
Sep 18 16:46:05 prs-sun-20-nyth-phitx     Fault_PC 0x10032934
Sep 18 16:46:05 prs-sun-20-nyth-phitx unix: [ID 836849 kern.notice]
Sep 18 16:46:05 prs-sun-20-nyth-phitx ^Mpanic[cpu8]/thread=30002e8f020:
Sep 18 16:46:05 prs-sun-20-nyth-phitx unix: [ID 886457 kern.notice] [AFT1] errID
0x00001bc0.06e19b00 BERR
Error(s)
Sep 18 16:46:05 prs-sun-20-nyth-phitx     See previous message(s) for details
Sep 18 16:46:05 prs-sun-20-nyth-phitx unix: [ID 100000 kern.notice]
Sep 18 16:46:05 prs-sun-20-nyth-phitx genunix: [ID 723222 kern.notice]
000002a1004e4e10 SUNW,UltraSPARC-III:cpu_aflt_log+44c (2a1004e4ece, 10145b38,
10145b10, 0, 2a1004e5058,
2a1004e4f1b)
```

Analysis

In this case we have a bus error detected by CPU 8 while trying to talk to physical address 0x00000630.0d806330. Note that the fault occurred in the kernel (privileged mode). The fault PC, while not exact (due to the async nature of bus error traps), at least gives us an idea as to where in the kernel the problem occurred. Examining the core dump and stack trace will give us a better idea.

This domain has 4 CPUs (8, 9, 10, and 11), on one system board.

Decoding the upper bits of the AFAR will tell us who CPU 8 was talking to when the BERR occurred:

0x630 == 0b0110.0011.0000

Breaking this down into fields:

[1] [1] [0.00] [11.000] [0]

The last two fields give us the AID 0x18 and the PCI leaf (in this case, Leaf B)

Knowing that the system is a 0x3800?, and contains a 4 slot cPCI, we can derive the actual physical slot or slots. Per SRDB 28506 this would be IB6, slot #2.

According to the prtdiag output, this card is:

CODE EXAMPLE 2-13 PRTdiag Output

```
 Bus  Max
            IO   Port Bus       Freq Bus  Dev,
FRU Name    Type  ID  Side Slot MHz  Freq Func State Name
Model
----------  ---- ---- ---- ---- ---- ---- ---- ----- ------------------------
-------  ----------------------
/N0/IB6/P0  cPCI  24   B    2   33   33  1,0  ok   pci-pci1011,46.1/pci108e,1000
pci-bridge
/N0/IB6/P0  cPCI  24   B    2    33   33  0,0  ok    pci108e,1000-pci108e,1000.1
/N0/IB6/P0  cPCI  24   B    2   33   33  0,1  ok    SUNW,hme-pci108e,1001
SUNW,cheerio
/N0/IB6/P0  cPCI  24   B    2    33   33  4,0  ok    SUNW,isptwo-pci1077,1020/sd
(blo+ QLGC,ISP1040B
(blo+ QLGC,ISP1040B
```

Analysis

So we had a bus error between CPU 8 and the hme card in IB6, slot #2. The three likely suspects are either SB2 (containing CPU 8), IB6, or the hme card. Further errors might help us choose which one is the likely problem.

Suspect FRUs

- SB2
- IB6
- hme card

CODE EXAMPLE 2-14 Panic Incident 2

```
Sep 18 17:14:01 prs-sun-20-nyth-phitx SUNW,UltraSPARC-III: [ID 193459
kern.warning] WARNING: [AFT1] Bus Error (BERR) Event on CPU10 Privileged Data
Access at TL=0, errID 0x00000167.482a5b10
Sep 18 17:14:01 prs-sun-20-nyth-phitx     AFSR 0x00000800<BERR>.00000000 AFAR
0x00000630.0d807010
Sep 18 17:14:01 prs-sun-20-nyth-phitx     Fault_PC 0x10032934
...
```

Analysis

This is exactly the same as incident 1, only a different CPU (10) is involved (but same IB board). Note that the fault PC is also the same. The AFAR, while slightly different, still points to the same Schizo and PCI slot.

Suspect FRUs

- SB2
- IB6
- hme card

CODE EXAMPLE 2-15 Panic Incident 3

```
Sep 19 13:54:47 prs-sun-20-nyth-phitx SUNW,UltraSPARC-III: [ID 856458
kern.warning] WARNING: [AFT1] Bus Error (BERR) Event on CPU11 Privileged Data
Access at TL=0, errID 0x0000124b.09aa4a90
Sep 19 13:54:47 prs-sun-20-nyth-phitx     AFSR 0x00000800<BERR>.00000000 AFAR
0x00000630.0d807010
Sep 19 13:54:47 prs-sun-20-nyth-phitx     Fault_PC 0x10032934
...
```

Analysis

This is exactly the same as incident 1, only a different CPU (10) is involved (but same system board). Note that the fault PC is the same.

CODE EXAMPLE 2-16 Panic Incident 4

```
Sep 20 02:23:35 prs-sun-20-nyth-phitx SUNW,UltraSPARC-III: [ID 671039
kern.warning] WARNING: [AFT1] Bus Error (BERR) Event on CPU9 Privileged Data
Access at TL=0, errID 0x00001615.d76736a0
Sep 20 02:23:35 prs-sun-20-nyth-phitx     AFSR 0x00000800<BERR>.00000000 AFAR
0x00000630.0d807010
Sep 20 02:23:35 prs-sun-20-nyth-phitx     Fault_PC 0x10032934
...
```

Analysis

This is exactly the same as incident 1, only a different CPU (CPU 9) is involved (but same system board). Note that the fault PC is the same.

We could look at incidents 5-8, but they are essentially the same as 1-4. Since we have all 4 CPUs reporting the same problem, one might assume that the CPU (or system board) is not likely the problem. Thus, IB6 and the hme card are the two likely suspects.

We should also have core dumps from the above 8 panics. Examination of one of them shows the stack trace is in the hme driver and the kernel was talking to hme instance #1. Since we had already isolated the problem to a specific slot, analysis of the corefile isn't necessary.

However, we still have two more incidents to examine.:

Note – The following incident is out of time order.

CODE EXAMPLE 2-17 Panic Incident 9

```
Sep 20 06:32:11 prs-sun-20-nyth-phitx pcisch: [ID 831440 kern.warning] WARNING:
pcisch-0: PCI fault log start:
Sep 20 06:32:11 prs-sun-20-nyth-phitx pcisch: [ID 619209 kern.notice] PCI
excessive retry error
Sep 20 06:32:11 prs-sun-20-nyth-phitx pcisch: [ID 630226 kern.notice] PCI error
ocurred on device #6
Sep 20 06:32:11 prs-sun-20-nyth-phitx pcisch: [ID 120591 kern.notice] dwordmask=
0 bytemask=f
Sep 20 06:32:11 prs-sun-20-nyth-phitx pcisch: [ID 607383 kern.notice] pcisch-0:
PCI primary error (8):
Sep 20 06:32:11 prs-sun-20-nyth-phitx pcisch: [ID 144556 kern.notice] Excessive
Retries
Sep 20 06:32:11 prs-sun-20-nyth-phitx pcisch: [ID 259679 kern.notice] pcisch-0:
PCI secondary error (0):
Sep 20 06:32:11 prs-sun-20-nyth-phitx pcisch: [ID 467665 kern.notice] pcisch-0:
PBM AFAR 0.0d802000:
Sep 20 06:32:11 prs-sun-20-nyth-phitx pcisch: [ID 908581 kern.notice] pcisch-0:
PCI config space error status (280):
Sep 20 06:32:11 prs-sun-20-nyth-phitx pcisch: [ID 141464 kern.notice] pcisch-0:
PCI fault log end.
Sep 20 06:32:11 prs-sun-20-nyth-phitx unix: [ID 836849 kern.notice]
Sep 20 06:32:11 prs-sun-20-nyth-phitx ^Mpanic[cpu8]/thread=2a10001fd40:
Sep 20 06:32:11 prs-sun-20-nyth-phitx unix: [ID 578303 kern.notice] pcisch-0:
PCI bus 2
error(s)!
```

Analysis

Since this is a PCI error, we will be covering this and other PCI errors later in detail. In summary this is an excessive retry error between IB6 (Schizo #0) and the hme card in IB6 slot #2.

CODE EXAMPLE 2-18 System Error

```
prs-sun-20-nyth-phitx-sc0:A> showlogs
Sep 22 03:09:35 prs-sun-20-nyth-phitx-sc0 Domain-A.SC: [ID 974093 local0.crit]
Domain A has a SYSTEM ERROR
Sep 22 03:09:35 prs-sun-20-nyth-phitx-sc0 Domain-A.SC: [ID 608736 local0.error]
/N0/IB6 encountered the first error
Sep 22 03:09:35 prs-sun-20-nyth-phitx-sc0 Domain-A.SC: [ID 159164
local0.error] RepeaterSbbcAsic reported first error on /N0/IB6
Sep 22 03:09:35 prs-sun-20-nyth-phitx-sc0 Domain-A.SC: [ID 222962 local0.error]
/partition0/domain0/IB6/bbcGroup0/sbbc0:
    ErrorStatus[0x80] : 0x80000000
                  ErrSum [31:31] : 0x1
>>> PCIErrorStatus[0xe0] : 0x00208020
                 AccSIll [21:21] : 0x1
                      FE [15:15] : 0x1
                    SIll [05:05] : 0x1 Slave port detected an illegal access

Sep 22 03:09:35 prs-sun-20-nyth-phitx-sc0 Domain-A.SC: [ID 253130 local0.crit]
Domain A is currently paused due to an error.  This domain must be turned off
via "setkeyswitch off" to recover
```

Analysis

According to the troubleshooting manual, there is no debugging information for this type of error. However, we do know that the problem was reported by IB6 and that it is a PCI error. As we will see later (see SBBC errors later), the SBBC is connected to Leaf B of Schizo #0 on I/O boards. So, this problem could be related to the incidents about (it is at least consistent).

Summary

All of the above instances (1-9), indict in one way or another, the PCI card in slot 2 of IB6.

Note – According to the case log, the IB6 PCI car was replaced and the case was closed.

Suspect FRU

- PCI card in slot 2 of IB6

2.2.5 Remaining CPU Errors

The remaining errors that a CPU can report are as follows.

- PERRProtocol Error
- IERRInternal Error
- ISAPIncoming System Address Parity Error

All of these errors will cause the a CPU to assert its ERROR pin and result in a SYSTEM ERROR, so you will never see these in a Solaris Panic message. All of these errors will be reported by the Repeater SBBC ASIC.

Note – When EMU or TUE errors occur on a CPU, they will also result in a SYSTEM ERROR. Also, there is a difference between EMU (the AFSR bit) and EMU (the external memory unit).

PERR is a System interface protocol error while IERR is an internal processing error. Errors that cause either a PERR or IERR are listed in the EESR or External Memory Unit (EMU), Error Status Register. This register is only accessible via JTAG, but is displayed and decoded after a system error. There are 77 unique error bits as reported in tables P-17 through P-28 of the SPARC V9 JPS1 Implementation Supplement.

IERR is an Internal error reported by the USIII. In all cases (regardless of which bit is set in the EMU), the suspect component is the CPU (and thus the FRU is the reporting SB board).

ISAP is an incoming system address parity error as seen by a CPU or Schizo. If this, error is seen, look for additional errors reported by the L1 AR, L2 AR etc. and use these to trace the problem to it's source. If this is the only error you see, the suspect FRU is the reporting board. See section on the AR ASIC for information and a diagram.

Note – For debugging PERRs: on Cheetah+, the AFAR can only be used for CPQ_TO, NCPQ_TO, and TID_TO. On Cheetah, the AFAR can't be used for PERRs as it is not valid.

Caution – Beware of bug ID 4410524 which incorrectly reports the EMU mappings.

TABLE 2-9 is a listing of EMU error bits excluding IERR and ISAP.

TABLE 2-9 EMU Errors (excluding IERR and ISA)

Error bit	Description
Parity and ECC errors	
S_PERR	Parity error on system address bus (copy of ISAP field of the AFSR)
M_ECC	Uncorrectable ECC on system data. Copy of Ored value of UE and IVU of the AFSR
E_ECC	Uncorrectable ECC on system data. Copy of Ored value of UE and IVU of the AFSR
MT_ECC	Uncorrectable ECC error on Mtag value. Copy of EMU field of the AFSR
MC Internal Errors	
MQ_OV	Memory controller backing queue overflows after PauseOut is asserte
System Bus Prototcol Error-Data	
UDT	Undefined DtransID
UTT	Undefined TtransID
MTARG	Multiple TargetID issued for same write transaction
UDG	Unexpected DtransID grant
UTG	Unexpected TargetID, TtransID grant
System Bus Prototcol Error Transaction	
USC	Undefined system bus command
CPQ_TOCPQ	System bus timeout
NCPQ_TONCPQ	System bus timeout
WQ_TO	Write transaction timeout
TID_TOT	TargetID timeout
AID_LKA	TtransID leakage error
CPQ_OVCPQ	Overflows after PauseOut is asserted
ORQ_UFORQ	Overflows after PauseOut is asserted
HBM_CONHBM	Mode contention
HBM_ERRHBM	Mode error
Snoop Result Errors	
RTS_SE	Local RTS Shared with Error
RTO_NDE	Local RTO no data and SharedIn = 0

TABLE 2-9 EMU Errors (excluding IERR and ISA) *(Continued)*

Error bit	Description
RTO_WDE	Local RTO wait data with SharedIn = 1
Mtag Errors	
SSM_MT	Mtag != gM in non-SSM mode
SSM_URT	Unexpected remote transaction (R_*) in non-SSM mode
SSM_URE	Unexpected reissued transaction from SSM device
SSM_IMT	SSM_IMTIllegal Mtag on returned data

2.2.5.1 Points to Remember When Debugging PERRs

One question still remains. Is there is any way to diagnose a WQ_TO? Presently, it is undiagnosable, as is every PERR detected by a processor. This is also true of UDT and SSM_URT errors.

Some guesses:

- MT_ECC should result from an EMU event in a CPU's AFSR (covered earlier).

  Hopefully, you can look at L1 DX error messages to locate the source of the bad Mtag ECC.

- It is not clear what error a TUE would generate, but you should see the AFSR value dumped.

  In any case, the component at fault would be the CPU reporting the error. Thus the FRU is the SB board reporting the error.

2.3 Schizo Errors

Since Schizo's job is to translate bus traffic between Safari and PCI, it can report errors on either side. This can simplify the debugging process.

- Safari-Side (Repeater / Centerplane) errors
- PCI-side errors

This section excludes CE and UE errors reported by Schizo which are covered previously. Possible Schizo (non-Excalibur mode) errors on the Safari side are listed in TABLE 2-10:

TABLE 2-10 Safari-Side (Repeater / centerplane) Schizo Errors

Error Bit	Description
Bad_Safari_Cmd	Unrecognized Safari command received
SSM_Disabled	Safari SSM command received and SSM mode not set
Bad_MIT-A-Cmd	Unrecognized command received from internal PCI-A leaf
Bad_MIT-B-Cmd	Unrecognized command received from internal PCI-B leaf
Safari_CIQ_Timeout	Transaction at head of CIQ longer than timeout interval
Safari_LPQ_Timeout	Transaction at head of LPQ longer than timeout interval
Safari_SFPQ_Timeout	Transaction at head of SFPQ longer than timeout interval
Safari_UFPQ_Timeout	Transaction at head of UFPQ longer than timeout interval
Safari_Addr_Par_Err	Safari Address Parity Error
Safari_Unmapped_Err	MappedIn snoop input not seen from transaction initated by Schizo
Safai_BusError_Dstat	Schizo received a Safari data packet for a non-cacheable DMA read indicating a bus error
Safai_Timeout_Dstat	Schizo received a safari data packet for a non-cacheable DMA read indicating a Timeout
Safai_Illegal_Dstat	Schizo received a safari data packet with an illegal Dstat value

Depending upon how the Safari Error Control / Interrupt registers are programmed, the response to any of these errors would be either an interrupt to Solaris and / or the assertion of S_ERROR_L.

If an interrupt is generated, it should call the Solaris function `cb_buserr_intr` (defined in `usr/src/uts/sun4u/io/pci/pcisch.c`), generating the following output:

CODE EXAMPLE 2-19 Safari Bus Error

```
368         sprintf(buf, "Safari bus error: "
 369                 "CSR=%016llx ErrCtrl=%016llx\n"
 370                 "IntrCtrl=%016llx ErrLog=%016llx\n"
 371                 "ECC_Ctrl=%016llx\n"
 372                 "UE_AFSR=%016llx UE_AFAR=%016llx\n"
 373                 "CE_AFSR=%016llx CE_AFAR=%016llx\n",
 374                         csr, err, intr, elog, ecc, ue_afsr, ue_afar,
 375                         ce_afsr, ce_afar);
 376          prom_printf(buf);
377         cmn_err(CE_PANIC, buf);
```

However, I can find no examples of any such panics. See bug #4414836.

If the Schizo asserts S_ERROR_L, I would assume the SYSTEM ERROR resulting would look very similar to what happens when a CPU gets an address parity error. following are a few examples which look like they may be candidates:

2.3.0.1 Example of SBBC `ErrorStatus/SafErr` on IB Board

This example is from a single segmented Sun Fire 6800 system. The source is Radiance Case #10116210.

CODE EXAMPLE 2-20 SBBC `ErrorStatus/SafErr` on IB Board

```
eqmfire2:A> showlogs
Aug 26 11:12:03 eqmfire2 Domain-A.SC: [ID 974093 local0.crit] Domain A has a
SYSTEM ERROR
Aug 26 11:12:04 eqmfire2 Domain-A.SC: [ID 608736 local0.error] /N0/IB6
encountered the first error
Aug 26 11:12:04 eqmfire2 Domain-A.SC: [ID 159164 local0.error] RepeaterSbbcAsic
reported first error on /N0/IB6
Aug 26 11:12:05 eqmfire2 Domain-A.SC: [ID 786555 local0.error]
/partition0/domain0/IB6/bbcGroup0/sbbc0:
>>> ErrorStatus[0x80] : 0x80008100
                     FE [15:15] : 0x1
                 ErrSum [31:31] : 0x1
                 SafErr [09:08] : 0x1 Fireplane device asserted an error

Aug 26 11:12:06 eqmfire2 Domain-A.SC: [ID 253130 local0.crit] Domain A is
currently paused due to an error.  This domain must be turned off via
"setkeyswitch off" to recover
```

Analysis

Why doesn't the SC report the value of the Safari Error Log Register? Without this we cannot tell which error type above could have caused this error. For CPU errors, this is done by CPU Safari agent.

Suspect FRU

- IB6

2.3.1 Schizo PCI-Side Errors

These errors are per leaf so there are two of these registers per Schizo.

Note – SBH and MMU can be disabled, as can all of the bits. However, checking the sources (`usr/src/uts/sun4u/io/pci/pci_space.c`) shows that they all seem to be enabled: uint_t ecc_error_intr_enable = 1; uint_t pci_sbh_error_intr_enable = (uint_t)-1; uint_t pci_mmu_error_intr_enable = (uint_t)-1:

TABLE 2-11 Schizo PCI-Side Errros

Error bits	Description
PBM Control/Status Register	
PCI_TTO_ERR	PCI TRDY# Timeout Error, detected during any PIO by Schizo
PCI_RTRY_ERR	Excessive Retry error during any PIO
PCI_MMU_ERR	An IOMMU error is detected during DMA
PCI_SBH_ERR	A streaming byte hole error during DMA write
PCI_SERR	SERR asserted on PCI bus
PCI AFSR (Primary)	
P_MA	Master Abort error
P_TA	DTarget Abort error
P_RTRY	Excessive retry error
P_PERR	Data parity error
P_TTO	TRDY# timeout error
P_UNSTABLE	BUS_UNSTABLE error
PCI AFSR (Secondary)	
	Same as above for primary register
PCI Status Register	
DPE	Set if PBM detects a parity error
SSE	Set if PCB signalled a system error (occurs if the PBM detects a PCI address parity error or another device asserts SERR#)
RMA	Set if PBM receives a master abort
TMA	set if PBM receives a target abort
STA	Set if PBM generates a target abort

Per Error handling table:

- PCI System Error SERR?
- PCI Address Parity Error PERR?
- PCI Master Abort - x

- PCI Target Abort - x
- PCI Retry Timeout x
- PCI Retry Limit Error -
- PCI Send Data Parity Error
- PCI Receive Data Parity Error
- IOMMU Translation Error

Helpful information:

- Instance # - will tell you which IB and range of slots involved (from 3 to 1)
- PCI AFAR (if appropriate) might tell you specifically which card?
- Slot number in register might tell you which card?
- Otherwise, for each of these we can isolate: the IB or the card, or interference from other cards, SBBC ? If not, there is no point in.

Note – Bus errors and Timeouts (to a CPU) actually have data in the Schizo registers that could be read.

2.3.1.1 Example of PCI Excessive Retry Error

This example if from a single segmented Sun Fire 6800 System with SB0, SB3 and SB4 in Domain A. The source is America Radiance case #62667973.

CODE EXAMPLE 2-21 PCI Excessive Retry Error

```
Sep 20 06:32:11 prs-sun-20-nyth-phitx pcisch: [ID 831440 kern.warning] WARNING:
pcisch-0: PCI fault log start:
Sep 20 06:32:11 prs-sun-20-nyth-phitx pcisch: [ID 619209 kern.notice] PCI
excessive retry error
Sep 20 06:32:11 prs-sun-20-nyth-phitx pcisch: [ID 630226 kern.notice] PCI error
ocurred on device #6
Sep 20 06:32:11 prs-sun-20-nyth-phitx pcisch: [ID 120591 kern.notice]
dwordmask=0 bytemask=f
Sep 20 06:32:11 prs-sun-20-nyth-phitx pcisch: [ID 607383 kern.notice] pcisch-
0: PCI primary error (8):
Sep 20 06:32:11 prs-sun-20-nyth-phitx pcisch: [ID 144556 kern.notice] Excessive
Retries
Sep 20 06:32:11 prs-sun-20-nyth-phitx pcisch: [ID 259679 kern.notice] pcisch-
0: PCI secondary error (0):
Sep 20 06:32:11 prs-sun-20-nyth-phitx pcisch: [ID 467665 kern.notice] pcisch-
0: PBM AFAR 0.0d802000:
Sep 20 06:32:11 prs-sun-20-nyth-phitx pcisch: [ID 908581 kern.notice] pcisch-
0: PCI config space error status (280):
Sep 20 06:32:11 prs-sun-20-nyth-phitx pcisch: [ID 141464 kern.notice] pcisch-
0: PCI fault log end.
Sep 20 06:32:11 prs-sun-20-nyth-phitx unix: [ID 836849 kern.notice]
Sep 20 06:32:11 prs-sun-20-nyth-phitx ^Mpanic[cpu8]/thread=2a10001fd40:
Sep 20 06:32:11 prs-sun-20-nyth-phitx unix: [ID 578303 kern.notice] pcisch-0:
PCI bus 2
error(s)!
```

Analysis

This PCI panic was reported by CPU 8, but the problem is actually with one of the PCI busses. As this system has two I/O boats (IB6 and IB8) and multiple PCI cards, it is not immediately clear which FRUs are involved.

Analyzing each field in the error log in more detail:

- `pcisch-0` actually decodes as `drivername == pcisch` and `instance == 0`. To decode this information refer to SRDB 28263.
- `pcisch-0` therefore refers to instance 0 of the PCI Schizo driver (representing a single PCI leaf A or B). To map the instances, refer to the `/etc/path_to_inst file`. Since we don't have `/etc/path_to_inst` here, I assume that the I/O boards would be mapped in slot order(?). Instance numbers appear to always

map PCI leaf B first, then A, and in Schizo order (first 0 then 1). With this in mind, instance 0 should be IB6, Schizo #0, PCI leaf B. Given that on a cPCI this maps to physical slot 2 or 3, and we know that there is nothing in slot 3, we are talking about the same hardware indicted by the previous 8 instances.

- Alternatively, by looking at the core dump and extracting the actual device path, as follows, the exact slot number can be determined:

/SUNW,Sun-Fire/ssm@0,0/pci108e,8001@18,700000/pciclass,060940@1/SUNW,hme@0,1

- `18,700000` means IB6, Schizo #0, PCI leaf B.
- `pciclass,060940@1`, where the `@1` means device 1. Both pieces of information and the table in SRDB 28263 tell you that it is IB6, cPCI physical slot #2.
- PCI excessive retry error is extracted from the PCI control and status register as defined in the Schizo Spec in the table 22-34.
- PCI error ocurred on device #6 indicates that device 6 was the bus master at the time of the error; this is captured by the PCI control and status register as defined in table 22-34. As device #6 maps to the Schizo itself, this doesn't give much information.
- The entry: dwordmask=0 bytemask=f, is not important.
- PCI primary error (8) is the sub-field (value == 8) of the PCI AFSR reporting the primary error. Note primary error just means the first error to occur and the one that relates to the AFAR that is reported later. A secondary error may be captured, but the AFAR won't be captured a second time. The PCI AFSR is defined in the Schizo Spec in the table 22-36. However, the next line Excessive Retries is decoding the value of the sub-field (8) for us. Note that this is the same error that was reported in the PCI control and status register above.
- PCI secondary error (0) is the sub-field (value == 0) of the PCI AFSR reporting any secondary errors. As there aren't any, no additional information is printed.
- PBM AFAR 0.0d802000 is the 32-bit physical PCI address associated with the primary error above. Note that it is suspiciously close to the lower 32-bits of the Safari physical address reported in incidents 1-8.
- PCI config space error status (280) is reporting the value of the PCI status register? as defined in table 22-43.
  - 280 (assume hex?) doesn't have any of the error bits set:
  - 0x8000 == PBM parity
  - 0x4000 == PCI address parity error or another device asserted SERR
  - 0x2000 == PBM master abort
  - 0x1000 == PBM target abort received
  - 0x0800 == PBM target abort sent
  - 0x0100 == PCI parity error while PBM is bus master

  So, this doesn't give us any new information.
- PCI bus 2 error(s)! indicates that two error bits were found. The number of error bits is the sum of:
  - PBM control status reg error bits (1)
  - PCI AFSR primary error bits (1)
  - PCI AFSR secondary error bits (0)

- PCI status register error bits (0)

While a total of two bits were reported, there is only one error (bug)?

Summary

We have an excessive retry error between the Schizo and the hme card in IB6 slot #2.

Suspect FRU

- IB6

CHAPTER 3

Data Path Errors

3.1 Overview

A combination of ECC and parity is used to protect the data path. ECC is used as an end-to-end check and is generated once at the source and corrected (if necessary) at the destination (except for memory). ECC is used to detect and correct failures in the CPU cache SRAMs and/or memory DRAMS, and it is also used in the data path (detection only). Parity, on the other hand, is generated, checked, and regenerated several times through the data path to help isolate point-to-point failures in the interconnect, however, not all paths are protected by parity.

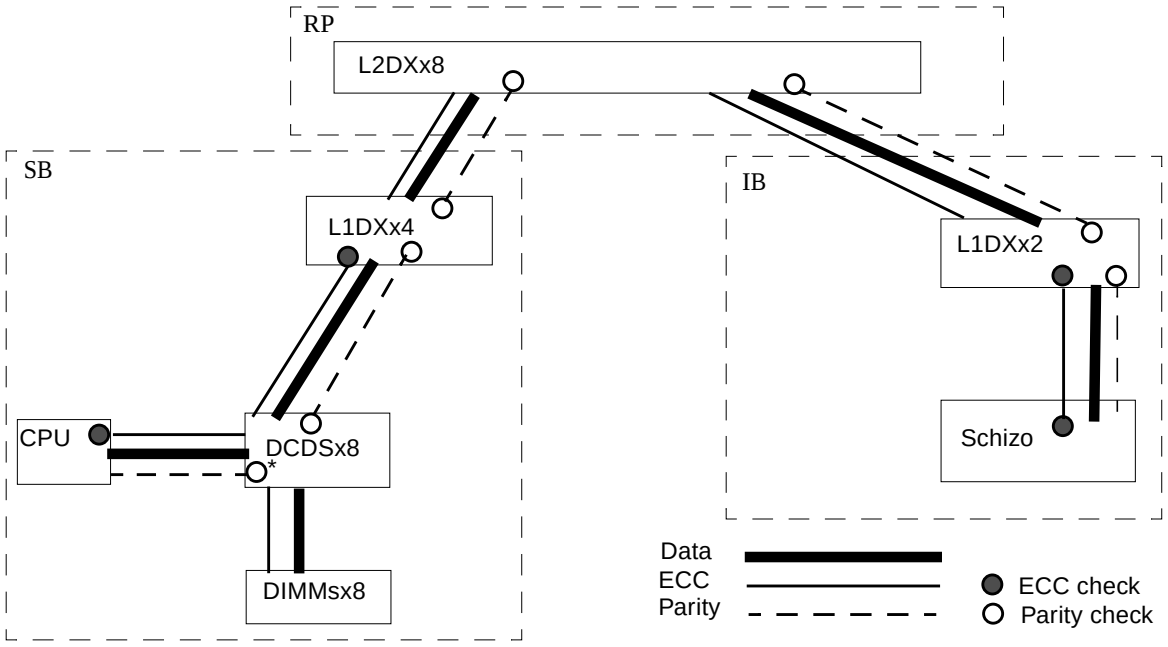
ECC limitations are:

- ECC can only handle one error per 128-bit block. Only single-bit errors and nibbles can be corrected. Not all multibit errors can be detected.
- Because it is end-to-end, it can be harder to locate the source of the error.
- Bad data travelling between a CPU and it's local memory will not be seen by any L1DX's.
- Bad data travelling between a CPU and it's pair (A/B or C/D) will only be seen once by the local L1DX's.

Parity limitations are:

- Parity circuitry can't correct data. It can only detect single bit flips (or odd #s of errors) per 18 bit block. A double bit error will not be seen.
- Some system data paths (L1DX to Schizo, and L1DX to CPU) are not protected by parity . This makes it possible for bit flips in these paths to be reported as ECC errors only. This can cause confusion if they are interpreted as DRAM ECC errors.

The following diagram illustrates data path ECC and parity protection as they relate to FRUs (from a system perspective):



* Parity is always CPU generated; DCDS always checks/reports errors

FIGURE 3-1 ECC Protected Data Path and Parity Detection

Note that some paths are not protected by parity (e.g. ECC only), namely:

- DX to Schizo
- DCDS to/from memory DIMMs.

A bit flip in these paths will result in ECC errors of various types (discussed later). It will be very important to be able to distinguish a bit flip in this path from a bit flip in a DIMM (which will also generate an ECC error).

A system component will never intentionally generate incorrect parity even if it is intentionally (in the case of a CPU or Schizo) or unintentionally (in the case of DRAMS) supplying bad ECC. Parity is calculated, based upon whatever ECC check bits are supplied by the sourcing device, regardless of whether it represents an ECC error or not. Thus, ECC errors originating from Ecache ECC errors or DRAM bit flips, will not be reported as parity errors.

ECC errors are reported to SC as they are detected in flight by L1DXs and by Solaris when they reach the destination. They can be recoverable. Parity errors are only reported only to the SC and are fatal (unfortunately).

3.2 Scope of Coverage

As follows:

3.3 ECC in the Data Path

The following diagram shows how ECC is generated once by a CPU, tested in-flight by each board it may pass through, and arriving at another CPU. It may be stored an indefinite period of time in DRAM without being tested.

Note that the following diagram shows multiple transactions (across three system boards).

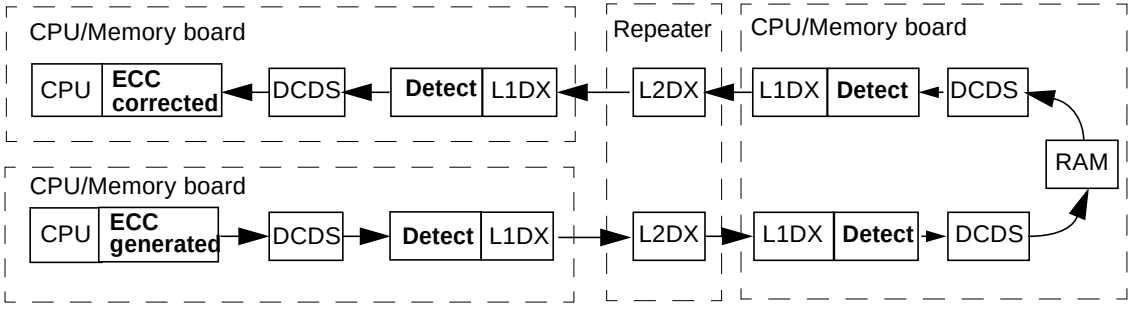


FIGURE 3-2 ECC Generation/Correction Path Example

3.3.1 DX and SDC ECC Registers

DX ECC errors detected by the data interconnect will have the format depicted in CODE EXAMPLE 3-1. Data pertaining to an error event will be captured by associated SDC and DX registers see TABLE 3-1 and TABLE 3-2 and are used to formulate the error message that is output as a result. Since SDC routes the transaction control signals it has a view (Status: `0x82f20000,` in example) of the agent ID (AID), the sequence # (SEQ#), etc. while the DX detects the actual ECC error.

CODE EXAMPLE 3-1 ECC Error Reported by DX ASIC

```
May 29 20:11:45 qads4-sc0 Domain-A.SC:
Status: 0x82f20000
Syndrome from DX2: 0x00000000
Syndrome from DX3: 0x80710000
Write transaction to Cheetah C: DTrans: 0x0f2 (AID=15, SEQ#=2)
ECC Syndrome: 0x071
MTag Syndrome: 0x0
```

TABLE 3-1 SDC ECC Status Register

Field	Bits	Description
P23_ECC_ERR	[31]	Port 2/3 logged an ECC error
Reserved	[30]	Reserved
P23_PORT	[29]	0 = Port 2; 1 = Port 3 Valid for incoming packets. See DX's P23_SYND_DIR. Use P23_DTRANSID to determine which port it is incoming for.
Reserved	[28:26]	Reserved
P23_DTARG	[25]	1 = write operation, 0 = read operation Port not valid for outgoing packets (leaving DX) when in IO mode.
P23_DTRANS_ID	[24:16]	DtransId[8:0] for the error. DtransId[8:4] is the AgentID for the destination of the data. The port is not valid for outgoing packets (leaving DX) when in IO mode.
P01_ECC_ERR	[15]	Port 0/1 logged an ECC error
Reserved	[14]	Reserved
P01_PORT	[13]	0 = Port 0; 1 = Port 1 Valid for incoming packets. See DX's P01_SYND_DIR. Use P01_DTRANSID to determine which port it is incoming for.

TABLE 3-1 SDC ECC Status Register *(Continued)*

Field	Bits	Description
Reserved	[12:10]	Reserved
P01_DTARG	[9]	1 = write operation, 0 = read operation Port not valid for outgoing packets (leaving DX) when in IO mode.
P01_DTRANSID	[8:0]	DtransId[8:0] for the error. DtransId[8:4] is the AgentID for the destination of the data. The port is not valid for outgoing packets (leaving DX) when in IO mode.

TABLE 3-2 DX Syndrome Register

Field	Bits	Description
P23_SYND_DIR	[31]	Port 2/3 logged syndrome direction. 0 = incoming (to DX); 1 = outgoing (from DX)
Reserved	[30]	Reserved
P23_MTAG_SYND	[29:26]	Port 2/3 MTAG ECC Syndrome
Reserved	[25]	Reserved
P23_ECC_SYND	[24:16]	Port 2/3 ECC Syndrome
P01_SYND_DIR	[15]	Port 0/1 logged syndrome direction. 0 = incoming (to DX); 1 = outgoing (from DX)
Reserved	[14]	Reserved
P01_MTAG_SYND	[13:10]	Port 0/1 MTAG ECC Syndrome
Reserved	[9]	Reserved
P01_ECC_SYND	[8:0]	Port 0/1 ECC Syndrome

The parameters for CODE EXAMPLE 3-1 are defined as follows:

- Status : (0x82f20000) is the SDC ECC Status Register.
- Syndrome from DX2: 0x00000000 is the DX2 syndrome register
- Syndrome from DX3: 0x80710000 is the DX3 syndrome register
- Dtrans is extracted from the SDC ECC status register, it contains the AID and SEQ fields.
- AID is the source or destination port (0-31) of a transaction
- SEQ# is the sequence number (1-15), use this to match up multiple transactions.

The Write transaction, ECC Syndrome and MTag Syndrome parameters are used to generate the SC interpretation. While the bits in these registers can be decoded, the information needed can be extracted from the messages output by the SC.

3.3.1.1 ECC Signalling Syndromes

Some ECC syndromes are reserved by hardware to indicate an intentional error, they are:

- 0x003
- 0x071
- 0x11c

The hardware does this to flag data that it knows is bad, but needs to pass on anyway. See Chapter 2, "Special ECC Signalling Syndromes" on page 2-9 for more information.

3.3.1.2 Agent IDs

CPU and Schizo Agent IDs are detailed in TABLE 3-3:

TABLE 3-3 CPU/Schizo Agent IDs

SB	CPU A (0)	CPU B (1)	CPU C (2)	CPU D (3)
SB0	0 (0x00)	1 (0x01)	2 (0x02)	3 (0x03)
SB1	4 (0x04)	5 (0x05)	6 (0x06)	7 (0x07)
SB2	8 (0x08)	9 (0x09)	10 (0x0a)	11 (0x0b)
SB3	12 (0x0c)	13 (0x0d)	14(0x0e)	15(0x0f)
SB4	16 (0x10)	17 (0x11)	18 (0x12)	19 (0x13)
SB5	20 (0x14)	21 (0x15)	22 (0x16)	23 (0x17)
IB	**Schizo 0**	**Schizo 1**		
IB6	24 (0x18)	25 (0x19)		
IB7	26 (0x1a)	27 (0x1b)		
IB8	28 (0x1c)	29 (0xd)		
IB9	30 (0x1e)	31 (0x1f)		

3.3.1.3 Port Numbers and Data Direction

In order to understand L1DX ECC errors, you must understand the limitations of L1DXs and SDCs capturing and reporting error information. SDC and DX registers have the capability to capture two errors: one for ports 0/1 and one for ports 2/3. Note that:

- Ports 0/1 represent (CPU A/B or memory A/B), or Schizo 0
- Ports 2/3 represent (CPU C/D or memory C/D), or Schizo 1

See following FIGURE 3-3 .

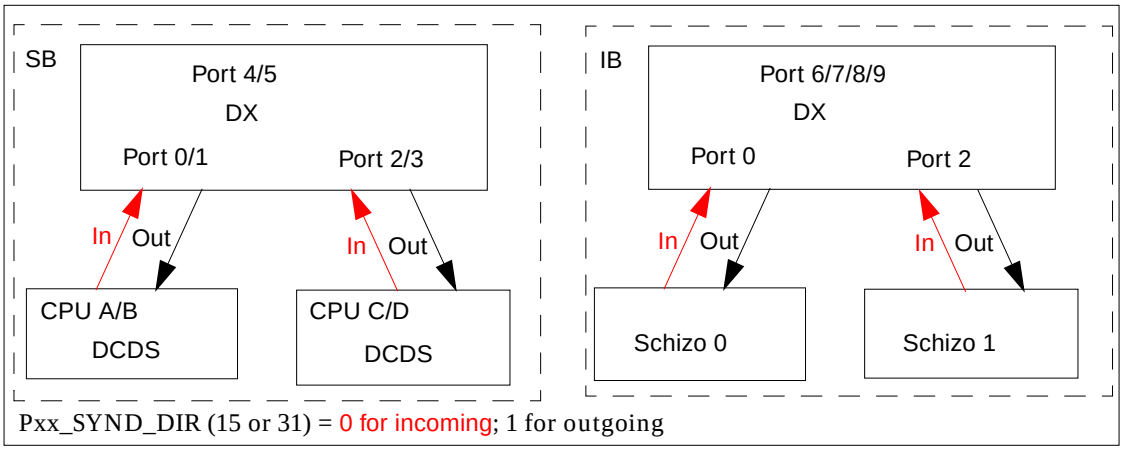


FIGURE 3-3 DX/CPU Port Numbers and Syndrome Direction for a System Board

The SDC status register will record (for each of the preceding port errors):

- Port number: either CPU A or B, or C or D), or Schizo 0 or 1
- Transaction type: read or write
- Target Aid: 0-31
- Sequence number: 1-15

Each error will have further information recorded in the DX register(s):

Note – A source of confusion can arise from misinterpreting the Pxx_SYND_DIR bits (bits 31 and 15 as shown in TABLE 3-2). These bits indicate the relative direction of incoming/outgoing data, with respect to the DX, and the onboard CPU or Schizo, see FIGURE 3-3. They do not indicate incoming/outgoing from the SB or the IB board assembly as is sometimes misinterpreted.

- Error direction: incoming to DX from onboard CPU/Schizo or DX outgoing to onboard CPU/Schizo
- ECC syndrome(s)
- Mtag syndrome(s)

Note that the port number is only valid for incoming transactions (e.g. from an onboard CPU to the DX). For outgoing transactions (e.g. from DX to onboard CPU/Memory) use the target aid from the syndrome to determine the true port. Unfortunately, this can confuse earlier versions of ScAPP (see Bug id#4477131).

Also note that incoming/outgoing is from the perspective of the DX with respect to ports 0/1 or 2/3 (e.g. DX to CPU is considered "outgoing"). Common sense seems to imply that the point of reference should be the system board reporting the error (e.g. going into or out of the board/FRU). This would reverse the direction.

The significance of the DX register numbers appearing in the error output (e.g. DX1, DX2 or DX3) will be explained later.

3.3.1.4 Data Width Information

The following data width information will prove useful in the discussions that follow:

- One quadword = 128 data bits + 9 ECC check bits + 3 Mtag data + 4 Mtag ECC = 144 bits
- One fireplane transaction = two quadwords
- One external cache line = 64 data bytes = four quadwords

3.3.1.5 CPU/Memory Board L1DX Registers

The syndrome reported for DX2 is really for DX2 and DX0 as they share ECC coverage of quadword 0 in cycle 0 of a two part fireplane transaction. Likewise the syndrome for DX3 represents DX1 and DX3 or quadword 1 in cycle 1.

To summarize:

- DX2 = error information for Quadword 0 (first half of transaction).
- DX3 = error information for Quadword 1 (second half of transaction).

You can have an error in either quadword or both. If you have an error in both quadwords, the direction bit (inbound/outbound) should be the same in both DX registers as they are both part of the same 2 cycle fireplane transaction.

The ECC / Mtag syndromes (in DX2 and DX3) may be different as the data path is 288 bits (2 Qwords) wide. In other words you can have independent errors in each quadword. However, the width of the USIII interface is 144 bits (1 Qword) wide. Thus, if the errors are the same it may indicate that the problem lies in the interface. Of course, you would then expect to see DCDS parity errors(?).

The width of the memory bus is 576?... Is this useful for mapping to DIMMs?

3.3.1.6 I/O Board L1DX Registers

The syndrome for DX1 is reported for I/O boards and represents DX0 and DX1 of a single quadword. Each quadword to/from a schizo is split in half and sent in two 72 bit cycles.

Consequently, a single bit error in the interface would lead to an uncorrectable Quadword doublebit error?

3.3.2 Interpreting "From", "For", and "To" Designations in L1DX ECC Error Messages

There are five variants of L1DX ECC error messages depending upon the system board type, transaction type, read or write, and transaction direction.

As an easy way to remember these, think of the "from" messages as being reported by the sourcing board. These messages contain both the source and destination information. The other variants ("for" and "to") are reported by the destination board and contain only destination information (plus they are susceptible to the bug listed below!).

1. Read transaction **from** Cheetah XXX (AID YYY)

- L1DX ECC Error detected *outgoing* (incoming to the local DX)
- Read data source: memory / CPU n*4+XXX
- Read destination: other CPU / Schizo YYY (can't be to memory)

2. Read transaction **for** Cheetah XXX (AID YYY)

   (See bug id #4477131, XXX can be "off by one", use AID instead)

- L1DX ECC Error detected *incoming* (outgoing to the local DX)
- Read data source: N / A (but see previous messages, if any)
- Read destination: CPU YYY

3. Write transaction **from** Cheetah XXX (AID YYY)

- L1DX ECC Error detected *outgoing* (incoming to the local DX)
- Write data source CPU n*4+XXX (can't be from memory)
- Write data destinationmemory / CPU / Schizo YYY

4. Write transaction **to** Cheetah XXX (AID YYY)

   (See bug id #4477131, XXX can be "off by one", use AID instead)

- L1DX ECC Error detected *incoming* (outgoing to the local DX)
- Write data source N / A (but see previous messages)
- Write data destination memory / CPU YYY

5. Fireplane transaction **to** SchizoM DTransID and DTarg not available

- L1DX ECC Error detected *incoming* (outgoing to the local DX)
- Data source: N / A (but see previous messages)
- Data destination: AID derived from IB # + Schizo #

Note – NOTE: There is no "Fireplane transaction from Schizo" message.

3.3.3 Interpreting ECC Syndromes

To decode an ECC syndrome from a CPU, Schizo or DX, use TABLE 3-4, together with the legend and example of table use.

TABLE 3-4 ECC Syndrome UltraSPARC III+ and Schizo

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	----	C0	C1	M2	C2	M2	M3	47	C3	M2	M2	53	M2	41	29	M
1	C4	M	M	50	M2	38	25	M2	M	33	24	M2	11	M	M2	16
2	C5	M	M	46	M2	37	19	M2	M	31	32	M	7	M2	M2	10
3	M2	40	13	M2	59	M	M2	66	M	M2	M2	0	M2	67	71	M
4	C5	M	M	43	M	36	18	M	M2	49	15	M	63	M2	M2	6
5	M2	44	28	M2	M	M2	M2	52	68	M2	M2	62	M2	M3	M3	M4
6	M2	26	106	M2	64	M2	M2	2	120	M	M2	M3	M	M3	M3	M4
7	116	M2	M2	M3	M2	M3	M	M4	M2	58	54	M2	M	M4	M4	M3
8	C7	M2	M	42	M	35	17	M2	M	45	14	M2	21	M2	M2	5
9	M	27	M	M2	99	M	M	3	114	M2	M2	20	M2	M3	M3	M
A	M2	23	113	M2	112	M2	M	51	95	M	M2	M3	M2	M3	M3	M2
B	103	M	M2	M3	M2	M3	M3	M4	M2	48	M2	M	73	M2	M	M3
C	M2	22	110	M2	109	M2	M	9	108	M2	M	M3	M2	M3	M3	M
D	102	M2	M	M	M2	M3	M3	M	M2	M3	M3	M2	M	M4	M	M3
E	98	M	M2	M3	M2	M	M3	M4	M2	M3	M3	M4	M3	M	M	M
F	M2	M3	M3	M	M3	M	M	M	56	M4	M	M3	M4	M	M	M
10	C8	M	M2	39	M	34	105	M2	M	30	104	M	101	M	M	4
11	M	M	100	M	83	M	M2	12	87	M	M	57	M2	M	M3	M
12	M2	97	82	M2	78	M2	M2	1	96	M	M	M	M	M	M3	M2
13	94	M	M2	M3	M2	M	M3	M	M2	M	79	M	69	M	M4	M
14	M2	93	92	M	91	M	M2	8	90	M2	M2	M	M	M	M	M4
15	89	M	M2	M3	M2	M	M3	M	M2	M	M3	M	M3	M	M	M3

TABLE 3-4 ECC Syndrome UltraSPARC III+ and Schizo

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
16	86	M	M2	M3	M2	M	M3	M	M2	M	M3	M	M3	M	M	M3
17	M	M	M3	M2	M3	M2	M4	M	60	M	M2	M3	M4	M	M	M2
18	M2	88	85	M2	84	M	M2	55	81	M2	M2	M3	M2	M3	M3	M4
19	77	M	M	M	M2	M3	M	M	M2	M3	M3	M4	M3	M2	M	M
1a	74	M	M3	M3	M	M	M3	M	M	M	M3	M	M3	M	M4	M3
1b	M2	70	107	M4	65	M2	M2	M	127	M	M	M	M2	M3	M3	M
1c	80	M2	M2	72	M	119	118	M	M2	126	76	M	125	M	M4	M3
1d	M2	115	124	M	75	M	M	M3	61	M	M4	M	M4	M	M	M
1e	M	123	122	M4	121	M4	M	M3	117	M2	M2	M3	M4	M3	M	M
1f	111	M	M	M	M4	M3	M3	M	M	M	M3	M	M3	M2	M	M

The following legend applies to TABLE 3-4.

Error bit	Description
----	No error
0 to 127	Single data bit error
C0 to C8	Single bit error on check bits 0 to 8
M2	Probable double bit error within a nibble
M3	Probable triple bit error within a nibble
M4	Probable quad bit error within a nibble
M	Multibit error
0x003, 0x71, 0x11c	Special ECC Signaling Syndromes (for more information see Chapter 2, “Special ECC Signalling Syndromes” on page 2-9).

See explanation example that follows for the use of this table:

3.3.3.1 How to Use Syndrome Table

As shown in CODE EXAMPLE 3-2, the ECC syndrome (Esynd) is 0x01d2. Applying this to TABLE 3-4, the last three digits are significent. The first two, 1 and d, represent the pertinent horizontal row number (1d) in the table while the third digit represents the table vertical column number, in this case 2. Thus, single bit 124 is in error.

CODE EXAMPLE 3-2 Interpreting ECC Syndrome

```
Oct 16 13:45:54 space SUNW,UltraSPARC-III: [ID 911686 kern.notice] NOTICE:
[AFT0]
Corrected system bus (CE) Event on CPU1 at TL=0, errID 0x0000020d.7edf1d70
Oct 16 13:45:54 space     AFSR 0x00000002<CE>.000001d2 AFAR 0x00000001.1959e090
Oct 16 13:45:54 space      Fault_PC 0x10024a6c Esynd 0x01d2 /N0/SB0/P2/B0/D0
J15300
Oct 16 13:45:54 space SUNW,UltraSPARC-III: [ID 874034 kern.notice] [AFT0] errID
0x0000020d.7edf1d70 Corrected Memory Error on /N0/SB0/P2/B0/D0 J15300 is
Persistent
Oct 16 13:45:54 space SUNW,UltraSPARC-III: [ID 628353 kern.notice] [AFT0] errID
0x0000020d.7edf1d70 Data Bit 124 was in error and corrected
```

Note that the bit in error that was corrected was also decoded and identified as 124 in a later line of the error message.

3.3.4 L1DX ECC Errors and SC Message Summary

- Bad data between CPU and it's Ecache results in no DX warning issued
- L1DXs on SB boards will issue warnings for memory requests on DIMMs with ECC issues
- L1DX on IB boards will issue warnings, for requests for / from Schizo that result in ECC issues
- L1DXs can only latch one error per CPU pair / Schizo (per board)
- For transactions between CPU pairs (A <-> B or C<-> D), errors will only be logged outgoing, not incoming, since DX can't capture second error.

  For transactions between CPU pairs or between boards, this isn't a problem.

For multiple error cases, involving the same source but different destinations (or vis versa), results can be confusing due to secondary errors that are not recorded. Thus, you must use the sequence #, syndrome, direction of error, source / destination AIDs to match up multiple L1DX ECC reports. Once reports are matched, the first instance of each error and identify of a FRU can be determined.

You can also reference the Solaris error messages (decoding of AFARs, IDs, transaction types, AFSRs, etc.). If there are DCDS parity errors, even more information is available.

The order of DX error messages in which they appear in the log is not consistent (i.e. they are not reported in time order for errors occurring at nearly the same time).

If the sequence numbers for a pair of ECC reports are different, then they cannot be part of the same transaction. If the sequence numbers are the same, then there is a good chance that they are part of the same transaction (but this is not certain).

3.3.5 Correctable ECC DIMM Error Propragating Through Interconnect

The following two examples are error messages output on the domain console and Solaris (var/adm/messages) for the same error event which is a correctable ECC error which was propagated through the interconnect.

Note – Disregard that the time stamps and domain names are different between the Solaris and SC messages as the faulty FRU was moved between systems.

CODE EXAMPLE 3-3 Domain Console Error Message

```
Oct 14 01:43:15 spectre-a Domain-A.SC:  /N0/SB0 reported ECC
error
Oct 14 01:43:15 spectre-a Domain-A.SC:  Status: 0xc1510000

Syndrome from DX2: 0x00000000
Syndrome from DX3: 0x01d20000

Read transaction from Cheetah C: DTrans: 0x151 (AID=21, SEQ#=1)
         ECC Syndrome: 0x1d2
         MTag Syndrome: 0x0

Oct 14 01:43:15 spectre-a Domain-A.SC: /N0/SB5 reported ECC error
Oct 14 01:43:15 spectre-a Domain-A.SC: Status: 0x00008151

Syndrome from DX2: 0x00000000
Syndrome from DX3: 0x000081d2

Read transaction for Cheetah A: DTrans: 0x151 (AID=21, SEQ#=1)
         ECC Syndrome: 0x1d2
         MTag Syndrome: 0x0
```

CODE EXAMPLE 3-4 Solaris Error Message

```
Oct 16 13:45:54 space SUNW,UltraSPARC-III: [ID 911686 kern.notice] NOTICE:
[AFT0]
Corrected system bus (CE) Event on CPU1 at TL=0, errID 0x0000020d.7edf1d70
Oct 16 13:45:54 space     AFSR 0x00000002<CE>.000001d2 AFAR 0x00000001.1959e090
Oct 16 13:45:54 space      Fault_PC 0x10024a6c Esynd 0x01d2 /N0/SB0/P2/B0/D0
J15300
Oct 16 13:45:54 space SUNW,UltraSPARC-III: [ID 874034 kern.notice] [AFT0] errID
0x0000020d.7edf1d70 Corrected Memory Error on /N0/SB0/P2/B0/D0 J15300 is
Persistent
Oct 16 13:45:54 space SUNW,UltraSPARC-III: [ID 628353 kern.notice] [AFT0] errID
0x0000020d.7edf1d70 Data Bit 124 was in error and corrected

Oct 16 13:45:54 space SUNW,UltraSPARC-III: [ID 244820 kern.info] [AFT2] errID
0x0000020d.7edf1d70 E$tag PA=0x0000000b.f8d9e080 does not match AFAR=
0x00000001.1959e080
Oct 16 13:45:54 space SUNW,UltraSPARC-III: [ID 801335 kern.info] [AFT2] errID
0x0000020d.7edf1d70 PA=0x0000000b.f8d9e080
Oct 16 13:45:54 space     E$tag 0x00000017.f1209201 E$state_2 Invalid
Oct 16 13:45:54 space SUNW,UltraSPARC-III: [ID 895151 kern.info] [AFT2] E$Data
(0x00) 0x00000002.000063d4 0xff253fba.ff3e2074 ECC 0x182
Oct 16 13:45:54 space SUNW,UltraSPARC-III: [ID 895151 kern.info] [AFT2] E$Data
(0x10) 0x00000000.00000000 0x00000000.00000000 ECC 0x000
Oct 16 13:45:54 space SUNW,UltraSPARC-III: [ID 895151 kern.info] [AFT2] E$Data
(0x20) 0x00000000.00000000 0xff3e3408.ffbefe90 ECC 0x084
Oct 16 13:45:54 space SUNW,UltraSPARC-III: [ID 895151 kern.info] [AFT2] E$Data
(0x30) 0xff3a0388.00002000 0x00000b00.ffbeffda ECC 0x196
Oct 16 13:45:54 space SUNW,UltraSPARC-III: [ID 929717 kern.info] [AFT2] D$ data
not available
Oct 16 13:45:54 space SUNW,UltraSPARC-III: [ID 929717 kern.info] [AFT2] D$ data
not available
```

3.3.5.1 Analysis

The domain console example depicts a pair of L1DX ECC messages with matching SEQ numbers, syndromes, etc. This is a read transaction issued by CPU B of SB5 from CPU/memory C of SB0. The syndrome 0x1d2 corresponds to bit 124, see TABLE 3-4. This is a correctable ECC syndrome. SB0 sourced the ECC syndrome and SB5 received the ECC syndrome.

The Solaris example message indicates a correctable ECC error was detected by CPU 1 (CPU B on SB0). The associated read maps to: /N0/SB0/P2/B0/D0 J15300 (controlled by CPU C on SB0). The syndrome and source matches the SC messages.

3.3.5.2 Summary

- The AFT2 Solaris messages from this case does not contain any useful information that we need to select a FRU for replacement.
- The combination of the SC and Solaris messages indicate that the single bit error likely originated from the indicated DIMM. Since the error was noticed by a CPU on a seperate system board, we have coraborating evidence from the SC messages.
- For this error event, there was a single bit ECC error on the affected DIMM and Solaris was able to correct it.

  For three or more CE reports against the same DIMM in a 24-hour period, see FIN I0760-1.

Suspect FRUs

- SB0: J15300
- SB0

3.3.6 Correctable ECC Error from a System Board to an I/O Assembly

CODE EXAMPLE 3-5 Correctable ECC Error from an SB to an IB

```
Aug 22 10:39:05 shqstbwd-sc0 Domain-A.SC [ID 542255 local0.error] /N0/SB0
reported ECC error
Aug 22 10:39:05 shqstbwd-sc0 Domain-A.SC [ID 700431 local0.error] Status:
0xc1880000
Syndrome from DX2: 0x01480000
Syndrome from DX3: 0x01090000
Read transaction from Cheetah C: DTrans: 0x188 (AID=24, SEQ#=8)
   DX2:
       ECC Syndrome: 0x148
        MTag Syndrome: 0x0
   DX3:
       ECC Syndrome: 0x109
       MTag Syndrome: 0x0

Aug 22 10:39:05 shqstbwd-sc0 Domain-A.SC [ID 354420 local0.error] /N0/IB6
reported ECC error
Aug 22 10:39:05 shqstbwd-sc0 Domain-A.SC [ID 498131 local0.error] Status:
0x00008000
Syndrome from DX1: 0x00008148
 Fireplane transaction to Schizo0: DTransID and DTarg not available
       ECC Syndrome: 0x148
       MTag Syndrome: 0x0

Aug 22 10:39:05 shqstbwd-sc0 Chassis-Port.SC [ID 138571 local0.error] /N0/SB0
reported first ECC error
Aug 22 10:39:05 shqstbwd-sc0 Chassis-Port.SC [ID 191816 local0.error]
Bad data read from a DIMM or cache controlled by //N0/SB0/P2

Aug 22 10:39:06 shqstbwd-sc0 Domain-A.SC [ID 542255 local0.error] /N0/SB0
reported ECC error
Aug 22 10:39:06 shqstbwd-sc0 Domain-A.SC [ID 958809 local0.error] Status:
0xc1890000
Syndrome from DX2: 0x00030000
Syndrome from DX3: 0x00030000

Read transaction from Cheetah C: DTrans: 0x189 (AID=24, SEQ#=9)
   ECC Syndrome: 0x003
   MTag Syndrome: 0x0
```

3.3.6.1 Analysis

In this case we have three L1DX ECC errors. The first two (reported by SB0 and IB6) appear to be related since they are reporting the same syndrome.

The first error appears to be from SBO CPU 2(C) issued by IB6 (schizo #0). SB0 reported two separate errors, DX2 reported one in quadword #0 (0x148), and DX3 reported one in quadword #1 (0x109)). The syndromes decode as follows:

- ECC syndrome 0x148 == data bit 90 (single bit)
- ECC syndrome 0x109 == data bit 30 (single bit)

Both are correctable. IB6 can only capture one error, and as such only reports the first quadword error (the second error is missed). Since SB0 sourced the error, and there are no DCDS parity errors, we would assume that the problem originated in CPU C's DIMMs.

Since there are no CE errors reported by Solaris, the above output is probably from the `showlogs` command.

Event #2 appears to be a read from CPU 2 (SB0) by IB6 (schizo #0) - again. Note that we don't have a matching error reported by IB6's L1 DX. The DX does not queue all outstanding ECC warnings. After the SC services the DX, pulls the logged data out, it resets the ECC Error register to allow other ECC errors to be logged.

In any case, the syndrome of 0x003 is an indication of either a double-bit error or an error intentionally forced by a Schizo as a result of a partial DMA write. However, for the later case to be true, the direction of the transaction is wrong. In this case, a CPU intentially delivering bad ECC would use the syndrome value of 0x71.

So, we can assume that this is a real double-bit error from either CPU C or it's memory.

3.3.6.2 Summary

The radiance case log indicates that SB0 was replaced (not sure about the DIMMs) and the case closed (at least as closely as I can translate French!)

3.3.6.3 Suspect FRUs

- One of SB0 DIMMs in CPU C's banks; there is no evidence to point to a specific DIMM
- SB0
- SB0 and all DIMMs in CPU C's banks

3.3.7 Example of Correctable ECC Error on SB Board

CODE EXAMPLE 3-6 Correctable ECC Error on SB Board

```
Aug 29 22:35:38 heslab-13c SUNW,UltraSPARC-III: CPU9 System bus data hardware
corrected
error: AFSR 0x00000002.00000023 AFAR 0x00000001.5157a280
Aug 29 22:35:38 heslab-13c     Syndrome 0x23 MemMod /N0/SB2/P2/B0/D1 on J15400,
Memory
controller 10
Aug 29 22:35:38 heslab-13c SUNW,UltraSPARC-III:      ECC Data Bit 46 was
corrected
Aug 29 22:35:38 heslab-13c SUNW,UltraSPARC-III: Softerror: Persistent ECC
Memory Error,
/N0/SB2/P2/B0/D1 on J15400, Memory controller 10
Aug 29 22:36:06 heslab-13 Domain-C.SC: /N0/SB2 reported ECC error
Aug 29 22:36:06 heslab-13 Domain-C.SC: Status: 0xc0928092
       Syndrome from DX2: 0x00238023
       Syndrome from DX3: 0x00000000

Read transaction for Cheetah A: DTrans: 0x092 (AID=9, SEQ#=2)
       ECC Syndrome: 0x023
       MTag Syndrome: 0x0

Read transaction from Cheetah C: DTrans: 0x092 (AID=9, SEQ#=2)
       ECC Syndrome: 0x023
      MTag Syndrome: 0x0
```

3.3.7.1 Analysis

A correctable ECC error is reported by CPU 9 against SB2, J15400. The error is recorded as persistent.

The syndrome from the SC (0x023) matches the Solaris message syndrome (0x23). The L1 DX ECC error was sourced from CPU B on SB2, and transfered to CPU C on SB2. It is likely that there is an ECC issue in the DIMM (J15400) indicated by Solaris.

3.3.7.2 Suspect FRUs

- SB2: J15400
- SB2

3.3.8 Example 1 of Correctable ECC Error Followed by SBBC ErrorStatus / DCDS Cheetah Parity Errors

The system that these examples came from is a single segment Sun Fire 6800 with Domain A containing SBs: 0, 3, and 4. This is the first of five examples of errors that occured on this system over a period of seven days. The source is America's Radiance Case #62601637.

CODE EXAMPLE 3-7 Example 1 of Correctable ECC Error Followed by SBBC ErrorStatus / DCDS Cheetah Parity Errors

```
NOTICE: [AFT0] Corrected system bus (CE) Event on CPU8 at TL=0, errID
0x0000001b.3c7f88a0
    AFSR 0x00000002<CE>.00000148 AFAR 0x00000000.ffc73150
    Fault_PC 0xf000f004 Esynd 0x0148 /N0/SB2/P1/B0/D2 on J14500, Memory
controller 9
[AFT0] errID 0x0000001b.3c7f88a0 Corrected Memory Error on /N0/SB2/P1/B0/D2 on
J14500, Memory controller 9 is Intermittent
[AFT0] errID 0x0000001b.3c7f88a0 Data Bit 90 was in error and corrected
Jul 31 21:20:01 itg-ctrl Domain-A.SC: Domain A has a SYSTEM ERROR
Jul 31 21:20:01 itg-ctrl Domain-A.SC: /N0/SB2 encountered the first error
Jul 31 21:20:01 itg-ctrl Domain-A.SC: RepeaterSbbcAsic reported first error on
/N0/SB2
Jul 31 21:20:01 itg-ctrl Domain-A.SC:
/partition0/domain0/SB2/bbcGroup1/sbbc1:
>>> ErrorStatus[0x80] : 0x80008010
                 DcdsErr [04:04] : 0x1 DCDS asserted an error
                      FE [15:15] : 0x1
                  ErrSum [31:31] : 0x1

Jul 31 21:20:01 itg-ctrl Domain-A.SC:
/partition0/domain0/SB2/bbcGroup1/cpuCD/dcds7:
>>> Cheetah0ErrorStatus[0x51] : 0x00008200
                      FE [15:15] : 0x1
                 C0DPerr [09:09] : 0x1 Cheetah 0 data error

Jul 31 21:20:01 itg-ctrl Domain-A.SC: Domain A is currently paused due to an
error.  This domain must be turned off via "setkeyswitch off" to recover
```

3.3.8.1 Analysis

In this case we have two errors, one reported by Solaris and one reported by the DCDS ASIC. The later is fatal and required the system to be power cycled.

Analyzing the first part of the message output by Solaris, we have a CE event detected by CPU #8 (SB 2, processor A). The physical address recorded maps to the bank of memory controlled by CPU #9 (SB2, processor 1, or B). This bank is reported as /N0/SB2/P1/B0/D2 or J14500. The syndrome for the CE is 0x148, but that really isn't important. Is the memory at fault? Note that we don't have any L1DX ECC errors.

The second part of the message is a DCDS data parity error message, which implicates CPUs C or D. Further, it is indicated that this is a Cheetah 0 data error. So we can assume this is really the fault of CPU C of SB 2. According to the troubleshooting manual, any DCDS errors are the result of the reporting FRU (in this case SB2).

Note: DCDS parity errors really shouldn't be fatal: see bug id: 4470487, since integrity is guaranteed by the ECC check as well. In this case we probably had a correctable ECC error as indicated by the Solaris message and could have recovered (even if SB 2 needs to be replaced).

Are the Solaris message and DCDS parity message related? They could be. Imagine that CPU 8 is requesting a cache line and instead of getting it from memory (owned by CPU 9), it has to get it from CPU 10's cache (since CPU 10 may have a dirty copy). This would result in a transaction between CPUs 10 and 8. However, this still doesn't explain why we didn't get any L1DX ECC messages. In any case, SB2 is probably at fault, while the Solaris message is at best ambiguous without further information.

3.3.9 Example 2 of System Error Followed by SBBC ErrorStatus and DCDS Cheetah Parity Errors

The system that these examples came from is a single segment Sun Fire 6800 with Domain A containing SBs: 0, 3, and 4. This is the second of five examples of errors that occured on this system over a period of seven days. The source is America's Radiance Case #62601637

CODE EXAMPLE 3-8 Example 2 of System Error Followed by SBBC ErrorStatus and DCDS Cheetah Parity Errors

```
Aug 02 16:26:53 itg-ctrl Domain-A.SC: Domain A has a SYSTEM ERROR
Aug 02 16:26:54 itg-ctrl Domain-A.SC: /N0/SB2 encountered the first error
Aug 02 16:26:54 itg-ctrl Domain-A.SC: RepeaterSbbcAsic reported first error on
/N0/SB2
Aug 02 16:26:54 itg-ctrl Domain-A.SC:
/partition0/domain0/SB2/bbcGroup1/sbbc1:
>>> ErrorStatus[0x80] : 0x80008010
                 DcdsErr [04:04] : 0x1 DCDS asserted an error
                      FE [15:15] : 0x1
                  ErrSum [31:31] : 0x1

Aug 02 16:26:54 itg-ctrl Domain-A.SC:
/partition0/domain0/SB2/bbcGroup1/cpuCD/dcds7:
>>> Cheetah0ErrorStatus[0x51] : 0x00008200
                      FE [15:15] : 0x1
                 C0DPerr [09:09] : 0x1 Cheetah 0 data error

Aug 02 16:26:54 itg-ctrl Domain-A.SC: Domain A is currently paused due to an
error.  This domain must be turned off via "setkeyswitch off" to recover
```

3.3.9.1 Analysis

This case is that same as that presented for example 1 only without the Solaris message.

3.3.9.2 Suspect FRU

- SB2

3.3.10 Example 3 of UE ECC Error Followed by SYSTEM ERROR / SBBC ErrorStatus / DCDS Cheetah Parity Errors

The system that these examples came from is a single segment Sun Fire 6800 with Domain A containing SBs: 0, 3, and 4. This is the third of five examples of errors that occured on this system over a period of seven days. The source is America's Radiance Case #62601637

CODE EXAMPLE 3-9 Example 3 of UE ECC Error Followed by SYSTEM ERROR / SBBC ErrorStatus and DCDS Cheetah Parity Errors

```
panic[cpu10]/thread=2a100309d40: [AFT1] errID 0x0000001b.5030c6c0 UE Error(s)
    See previous message(s)for details

panic[cpu9]/thread=2a10000bd40: syncing file systems...
panic[cpu10]/thread=2a100309d40: [AFT1] errID 0x0000001b.50c82a60 UE WDU
Error(s)
    See previous message(s) for details
skipping system dump - no dump device configured
Aug 04 13:18:04 itg-ctrl Domain-A.SC: Domain A has a SYSTEM ERROR
Aug 04 13:18:04 itg-ctrl Domain-A.SC: /N0/SB2 encountered the first error
Aug 04 13:18:04 itg-ctrl Domain-A.SC: RepeaterSbbcAsic report
ed first error on /N0/SB2
Aug 04 13:18:04 itg-ctrl Domain-A.SC:
/partition0/domain0/SB2/bbcGroup1/sbbc1:
>>> ErrorStatus[0x80] : 0x80008010
                DcdsErr [04:04] : 0x1 DCDS asserted an error
                     FE [15:15] : 0x1
                 ErrSum [31:31] : 0x1

Aug 04 13:18:04 itg-ctrl Domain-A.SC:
/partition0/domain0/SB2/bbcGroup1/cpuCD/dcds7:
>>> Cheetah0ErrorStatus[0x51] : 0x00008200
                     FE [15:15] : 0x1
                C0DPerr [09:09] : 0x1 Cheetah 0 data error

Aug 04 13:18:04 itg-ctrl Domain-A.SC: Domain A is currently paused due to an
error.  This domain must be turned off via "setkeyswitch off" to recover
```

3.3.10.1 Analysis

This case is the same as that presented for example 1 except the Solaris message is different. Solaris is indicating that we have both UE and Ecache ECC errors on CPU 10. Without the detailed information regarding the syndromes it is impossible to tell what really happened.

In any case, the DCDS error is unambiguous at blaming SB2.

3.3.11 Example 4 of ECC Error Followed by SYSTEM ERROR / SBBC ErrorStatus / DCDS Cheetah Bypass Errors

The system that these examples came from is a single segment Sun Fire 6800 with Domain A containing SBs: 0, 3, and 4. This is the fourth of five examples of errors that occured on this system over a period of seven days. The source is America's Radiance Case #62601637.

CODE EXAMPLE 3-10 Example 4 of ECC Error Followed by SYSTEM ERROR/SBBC ErrorStatus/DCDS Cheetah Bypass Errors

```
Aug 06 19:58:15 itg-ctrl Domain-A.SC: /N0/SB2 reported ECC error
Aug 06 19:58:15 itg-ctrl Domain-A.SC: Status: 0xc28b828b
Syndrome from DX2: 0x00108010
Syndrome from DX3: 0x01508150
Write transaction to Cheetah A: DTrans: 0x08b (AID=8, SEQ#=11)
        DX2:
                ECC Syndrome: 0x010
                MTag Syndrome: 0x0
        DX3:
                ECC Syndrome: 0x150
                MTag Syndrome: 0x0
Write transaction from Cheetah C: DTrans: 0x08b (AID=8, SEQ#=11)
        DX2:
                ECC Syndrome: 0x010
                MTag Syndrome: 0x0
        DX3:
                ECC Syndrome: 0x150
                MTag Syndrome: 0x0
Aug 06 19:58:24 itg-ctrl Domain-A.SC: Domain A has a SYSTEM ERROR
Aug 06 19:58:24 itg-ctrl Domain-A.SC: /N0/SB2 encountered the first error
Aug 06 19:58:24 itg-ctrl Domain-A.SC: RepeaterSbbcAsic reported first error on
/N0/SB2
Aug 06 19:58:24 itg-ctrl Domain-A.SC:
/partition0/domain0/SB2/bbcGroup1/sbbc1:
>>> ErrorStatus[0x80] : 0x80008010
                DcdsErr [04:04] : 0x1 DCDS asserted an error
                      FE [15:15] : 0x1
                  ErrSum [31:31] : 0x1
Aug 06 19:58:24 itg-ctrl Domain-A.SC:
/partition0/domain0/SB2/bbcGroup1/cpuCD/dcds7:
>>> Cheetah0ErrorStatus[0x51] : 0x00008080
                      FE [15:15] : 0x1
               S2CBypass [07:07] : 0x1 Cheetah 0 attempted to bypass Fireplane
while S2C0 fifo was not empty
Aug 06 19:58:24 itg-ctrl Domain-A.SC: Domain A is currently paused due to an
error.  This domain must be turned off via "setkeyswitch off" to recover
Aug 06 19:58:24 itg-ctrl Domain-A.SC: Domain watchdog timer expired.
```

3.3.11.1 Analysis

There are L1DX ECC errors, but no Solaris error messages. We also have a slightly different variation of the DCDS error (although that doesn't change the diagnosis).

The two L1DX ECC errors are clearly related since they have the same syndrome, sequence number, read/write type and are opposite directions. Looking at them together they say that CPU 10 issued a write to CPU 8 (or it's memory). Two ECC errors were detected in adjacent quadwords. So, since CPU 10 sourced the data (and the syndromes are not signalling), CPU 10 is at fault (again SB 2).

This at least corresponds nicely to the DCDS fault (also against CPU 10).

This is the same as the previous examples only there are four L1DX ECC messages representing two separate transactions. In each transaction, CPU 10 apparently sourced the bad data. Again, the DCDS failure points to SB2, CPU 10/C.

3.3.11.2 Summary

While there is missing information in some of the examples, all of the DCDS failures clearly point to SB2 (even CPU C). The radiance case log shows that the errors only stopped after SB2 was replaced. However, the DIMM implicated in example 1 was replaced first, but had no effect and the system continued to crash.

3.3.11.3 Suspect FRU

- SB2

3.3.12 Example 5 of ECC Error Followed by SYSTEM ERROR/SBBC ErrorStatus/DCDS Cheetah Parity Errors

The system that these examples came from is a single segment Sun Fire 6800 with Domain A containing SBs: 0, 3, and 4. This is the fifth of five examples of errors that occured on this system over a period of seven days. The source is America's Radiance Case #6260163.

CODE EXAMPLE 3-11 Example 5 of ECC Error Followed by SYSTEM ERROR/SBBC ErrorStatus/DCDS Cheetah Parity Errors

```
Aug 07 20:53:09 itg-ctrl Domain-A.SC: /N0/SB2 reported ECC error
Aug 07 20:53:09 itg-ctrl Domain-A.SC: Status: 0xc0818081
Syndrome from DX2: 0x01948194
Syndrome from DX3: 0x01948194
Read transaction for Cheetah A: DTrans: 0x081 (AID=8, SEQ#=1)
        ECC Syndrome: 0x194
        MTag Syndrome: 0x0
Read transaction from Cheetah C: DTrans: 0x081 (AID=8, SEQ#=1)
        ECC Syndrome: 0x194
        MTag Syndrome: 0x0
Aug 07 20:53:12 itg-ctrl Domain-A.SC: /N0/SB2 reported ECC error
Aug 07 20:53:12 itg-ctrl Domain-A.SC: Status: 0xc0818081
Syndrome from DX2: 0x010a810a
Syndrome from DX3: 0x00828082
Read transaction for Cheetah A: DTrans: 0x081 (AID=8, SEQ#=1)
        DX2:
                ECC Syndrome: 0x10a
                MTag Syndrome: 0x0
        DX3:
                ECC Syndrome: 0x082
                MTag Syndrome: 0x0
Read transaction from Cheetah C: DTrans: 0x081 (AID=8, SEQ#=1)
        DX2:
                ECC Syndrome: 0x10a
                MTag Syndrome: 0x0
        DX3:
               ECC Syndrome: 0x082
                MTag Syndrome: 0x0
Aug 07 20:53:12 itg-ctrl Domain-A.SC: Domain A has a SYSTEM ERROR
Aug 07 20:53:12 itg-ctrl Domain-A.SC: /N0/SB2 encountered the first error
Aug 07 20:53:12 itg-ctrl Domain-A.SC: RepeaterSbbcAsic reported first error on
/N0/SB2
Aug 07 20:53:12 itg-ctrl Domain-A.SC:
/partition0/domain0/SB2/bbcGroup1/sbbc1:
>>> ErrorStatus[0x80] : 0x80008010
                 DcdsErr [04:04] : 0x1 DCDS asserted an error
                      FE [15:15] : 0x1
                  ErrSum [31:31] : 0x1
Aug 07 20:53:13 itg-ctrl Domain-A.SC:
/partition0/domain0/SB2/bbcGroup1/cpuCD/dcds7:
>>> Cheetah0ErrorStatus[0x51] : 0x00008200
                      FE [15:15] : 0x1
                 C0DPerr [09:09] : 0x1 Cheetah 0 data error
Aug 07 20:53:13 itg-ctrl Domain-A.SC: Domain A is currently paused due to an
error.  This domain must be turned off via "setkeyswitch off" to recover
```

3.3.12.1 Analysis

This is the same as the previous example except that there are four L1DX ECC messages representing two separate transactions. In each transaction, CPU 10 apparently sourced the bad data. Again, the DCDS failure points to SB2, CPU 10/C.

3.3.12.2 Summary

While there is missing information in some of the examples, all of the DCDS failures clearly point to SB2 (even CPU C). The radiance case log shows that the errors only stopped after SB2 was replaced. However, the DIMM implicated in example 1 was replaced first. This DIMM replacement had no effect and the system continued to crash.

3.3.12.3 Suspect FRU

- SB2

3.4 Parity Detection in the Data Path

FIGURE 3-4 indicates a typical chain of data path parity checks by the DCDSs and L1/L2 DXs, from CPU to CPU, through memory. Parity, represented by dotted lines and arrows, is generated at the starting point, or the base of the arrows, while parity detection is represented by the tips of the arrows. The diagram shows two transactions, a write to memory from a CPU, followed by a read from memory by a CPU.

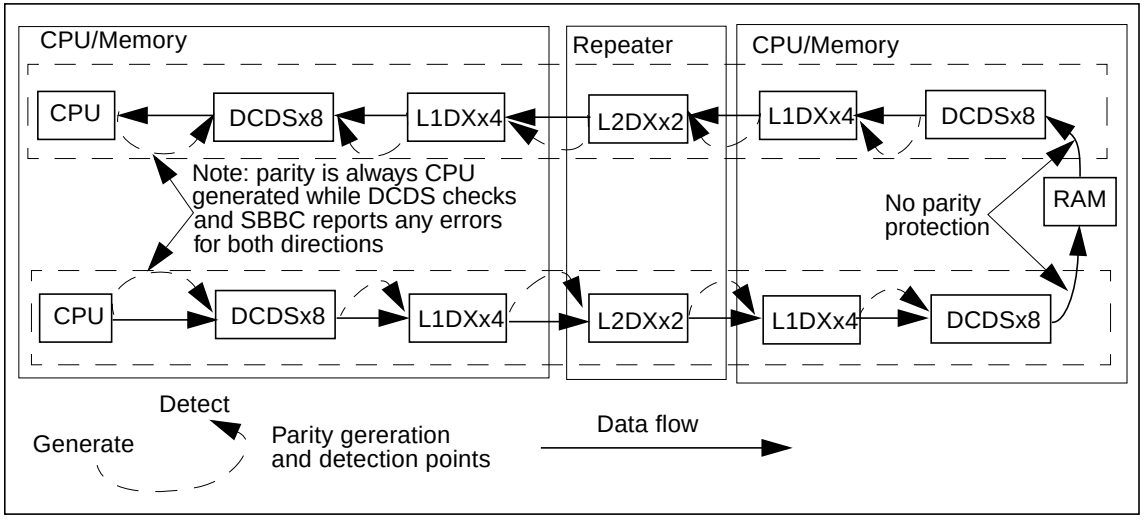


FIGURE 3-4 Data Path Parity Coverage From CPU to CPU Through Memory

3.4.1 DCDS Parity Errors

The DCDS ASICs can detect bad parity incoming from a CPU or from the L1 DX. As neither of these will normally generate bad parity, we know the fault lies within the system board. Since the 144 bits input to the DCDSs are bit sliced across eight DCDSs (18 bits per DCDS), a partial parity checking scheme is used. DCDS7 is the summation checker and will flag any errors to the associated SBBC ASIC which reports the errors.

3.4.2 L1DX Parity Errors

Parity error messages generated by system L1DXs identify the port (Safari Port) on which the error occured. An L1DX can detect bad parity incoming from the DCDS or from the centerplane (via the repeaters). DX ports 4/5 (SB boards) and 6-9 (IB board) route data from/to the interconnect, through DXs on repeater boards, see FIGURE D-1 in the S*un Fire 6800/4810/4800/3800 Systems Troubleshooting Guide*. Data is routed between DCDSs and the DXs through ports 0-3 (SB boards) or between Schizos and the DXs through ports 0/2 (IB boards).

If bad parity is detected incoming from the DCDS or Schizo, it is clearly a fault within the reporting board. You can tell if this is the case if the the parity error is associated with ports 0-3 (SB board) or ports 0/2 (IB board).

If bad parity is detected incoming from the centerplane (ports 4/5 on an SB board or ports 6-9 on an IB board), the fault may still be the board, it may be the centerplane, or it may be a repeater board. You can use the tables in Appendix D of the *Sun Fire 6800/4810/4800/3800 Systems Troubleshooting Manual* to isolate which repeater board. Also, further L2DX parity errors may help you locate the source.

3.4.2.1 Example of `Safari Port Error Status/SafPar` on SB Board

CODE EXAMPLE 3-12 `Safari Port Error Status/SafPar` on SB Board

```
Sep 27 08:32:53 ssminnow-sc0 Chassis-Port.SC: Domain A has a SYSTEM ERROR
Sep 27 08:32:52 ssminnow-sc0 Domain-A.SC: Domain A has a SYSTEM ERROR
Sep 27 08:32:53 ssminnow-sc0 Domain-A.SC: /N0/SB0 encountered the first error
Sep 27 08:32:53 ssminnow-sc0 Domain-A.SC: DxAsic reported first error on /N0/SB0
Sep 27 08:32:53 ssminnow-sc0 Domain-A.SC: /partition0/domain0/SB0/dx3:
>>> Safari Port Error Status 2[0x21] : 0x0000a000
              FirstError [15:15] : 0x1
                  SafPar [13:13] : 0x1 Fireplane Input bus parity error

Sep 27 08:32:54 ssminnow-sc0 Chassis-Port.SC: Domain A is currently paused due
to an error.
This domain will be powered  off and on via "setkeyswitch " to recover
```

Analysis

This error is against DX3 port 2 on SB0. SB0 DX ports 0 through 3 are connected to the DCDSs, see FIGURE D-1 in the S*un Fire 6800/4810/4800/3800 Systems Troubleshooting Guide*. Thus, this is a parity error against data input from the DCDS.

Suspect FRU

- SB0

3.4.2.2 Example of DX `Safari Port Error Status/ScU3IFPar` on SB Board (From Centerplane)

The following example is from a Sun Fire 6800 system.

CODE EXAMPLE 3-13 DX `Safari Port Error Status/ScU3IFPar` on SB Board

```
noname:A> showlog
Jul 12 11:10:01 noname Domain-A.SC: [ID 974093 local0.crit] Domain A has a SYSTEM
ERROR
Jul 12 11:10:01 noname Domain-A.SC: [ID 173191 local0.error] /N0/SB3 encountered
the first error
Jul 12 11:10:01 noname Domain-A.SC: [ID 200483 local0.error] DxAsic reported
first error on /N0/SB3
Jul 12 11:10:01 noname Domain-A.SC: [ID 772768 local0.error]
/partition0/domain0/SB3/dx2:
>>> Safari Port Error Status 4[0x23] : 0x00008004
               ScU3IFPar [02:02] : 0x1 Fireplane u3 input FIFO parity error
(ports 0 and 2 only)/Input parity error (other ports)
             FirstError [15:15] : 0x1

Jul 12 11:10:01 noname Domain-A.SC: [ID 253130 local0.crit] Domain A is
currently paused due to an error.  This domain must be turned off via
"setkeyswitch off" to recover.
```

Analysis

Ports 4 and 5 are the port connections used on SB DXs to connect, through the centerplane, to DXs on repeater boards. By referencing TABLE D-3 and FIGURE D-1 in the *Sun Fire 6800/4810/4800/3800 Systems Troubleshooting Guide*, it can be seen that port 4 of the SB3 DX2 (which reported the error) is connected to port 3 of DX0 on repeater RP1.

Suspect FRUs

- RP1
- SB3
- Centerplane

3.4.2.3 Example of DX `Safari Port Error Status/ScU3IFPar` on IB Board (From Centerplane)

The following error example is from a Sun Fire 4800 system:

CODE EXAMPLE 3-14 DX `Safari Port Error Status/ScU3IFPar` on IB Board

```
Sep 18 12:54:39 box1 Domain-A.SC: [ID 974093 local0.crit] Domain A has a SYSTEM
ERROR
Sep 18 12:54:39 box1 Domain-A.SC: [ID 687022 local0.error] RP0 encountered the
first error
Sep 18 12:54:39 box1 Domain-A.SC: [ID 556553 local0.error] DxAsic reported first
error on /N0/IB6
Sep 18 12:54:39 box1 Domain-A.SC: [ID 130920 local0.error]
/partition0/domain0/IB6/dx1:
>>> Safari Port Error Status 6[0x25] : 0x00048004
            AccScU3IFPar [18:18] : 0x1
               ScU3IFPar [02:02] : 0x1 Fireplane u3 input FIFO parity error
(ports 0 and 2 only)/Input parity error (other ports)
              FirstError [15:15] : 0x1

Sep 18 12:54:40 box1 Domain-A.SC: [ID 253130 local0.crit] Domain A is currently
paused due
to an error.  This domain must be turned off via "setkeyswitch off" to recover
```

Analysis

DX1 port 6 on board IB6 is reporting the error. By referencing TABLE D-3 and FIGURE D-2 in the *Sun Fire 6800/4810/4800/3800 Systems Troubleshooting Guide*, it can be seen that port 6 of the IB6 DX1 (which reported the error) is connected to port 6 of DX1 on repeater RP0. Since bad parity is detected incoming from the centerplane, the fault may be board, the centerplane, or the repeater board.

Suspect FRUs:

- RP1
- IB6
- Centerplane

3.4.3 L2DX Parity Errors (Repeater board)

An L2DX can detect bad parity incoming from the centerplane. In this case, the port #, repeater board #, and system type will allow you to determine which slot the bad parity came from, use the tables in Appendix D of the *Sun Fire 6800/4810/4800/3800 Systems Troubleshooting Manual*. As in previous L1DX cases, the fault may be with the reporting repeater board, in the centerplane, or the L1 board. Further L1DX parity errors may help locate the source.

3.4.4 DX ASIC FIFO RAM Parity Errors

All DXs (L1s and L2s) have the ability to detect parity errors in the FIFO ram that temporarily stores transit data through the DXs. In this case the fault is always with the reporting board.

You can tell the difference between an incoming parity error and a FIFO RAM parity error by the error bit reported during the SYSTEM ERROR dump. See the DX ASIC section for more information.

CHAPTER 4

Domain System Errors

Domain System errors are the result of an error detection process by the SC that produces an error event record for errors fatal to a domain's operation. For boards which are configured as part of a domain, these errors will be detected by the SC in the form of an interrupt whereas boards which are not part of a domain are polled. Domain System errors begin with a unique message string:

"**Domain A has a SYSTEM ERROR**"

This is followed by one or more error messages associated with the event. Since these messages are an indication that an error fatal to a domain's continued operation was detected, the messsage ends with the following:

Domain A is currently paused due to an error. This domain will be powered off and on via "setkeyswitch " to recover

See CODE EXAMPLE 4-1.

CODE EXAMPLE 4-1 Domain System Error

```
Jul 31 15:48:14 sp4-sc0 Domain-A.SC: ErrorMonitor: Domain A has a SYSTEM ERROR
Jul 31 15:48:14 sp4-sc0 Domain-A.SC: /N0/SB2 encountered the first error
Jul 31 15:48:15 sp4-sc0 Domain-A.SC: ArAsic reported first error on /N0/SB2
Jul 31 15:48:15 sp4-sc0 Domain-A.SC: /partition0/domain0/SB2/ar0:
>>> ConsolePortError[0x400] : 0x00008010
 CsILL [04:04] : 0x1 Console slave port detected an illegal access
 FE [15:15] : 0x1
Jul 31 15:48:15 sp4-sc0 Domain-A.SC: Domain A is currently paused due to an
error.  This domain will be powered off and on via "setkeyswitch " to recover
```

A domain SYSTEM ERROR can be caused by any of the following:

4.1 Error Capture and Propagation

System ASICs follow a standard error reporting procedure. Each ASIC can have multiple error registers. For each Safari and Console bus port that is connected to an ASIC, there is a corresponding error register. With the exception of the Interconnect Test Error register, all the error registers follow the same general format.

When an error event occurs, the SC will receive an interrupt, causing it to probe a board's error resisters. The information is collected on the SC by an Error Mask and Latch FPGA (Floating Programmable Gate Array). A second Glue FPGA, collects data concerning ECC events. Subsequently, the SC will output one or more messages concerning the SYSTEM ERROR event, based on the data found in these registers.

The following sections provide information on the first error (FE) bit as well as the Error Mask and Latch FPGA, also referred to as the Echip.

4.1.1 First Error (FE) Bit

ASICs have the ability to record which of its error registers (for example, Safari or Console port, etc.) saw the first error within the ASIC. When multiple errors occur simultaneously, multiple registers can set the FE bit. The FE bit is only valid for a single ASIC and has no relation to errors reported by other ASICs in the system.

The general format of the ASIC error registers is 32-bits divided into two 14-bit error fields. The first set of errors (all occurring within the same cycle) will be recorded in the lower 14 [0:14] bits. Subsequent errors will be recorded (and accumulated) in the upper 14 [16:31) bits.

Each error register's bit 15 is FE (First Error). The FE bit is set in the first error register in an ASIC that has an unmasked error logged. All subsequent errors will not set the FE bit in any other register in the ASIC, until the FE bit is cleared in all registers in that ASIC. If multiple registers log an error simultaneously, then the FE bit will be set in multiple registers.

4.1.2 Error Latch and Mask (FPGA) and Glue FPGA

ASIC and L1 DX ECC errors on SB or IB boards are recorded by Error Mask and Latch FPGAs and forwarded to the SC, see FIGURE 4-1. These FPGAs (also referred to as Echips) record which ASIC on a board reported the first error. ASIC errors on a repeater board are sent directly to the SC since they have no Echips.

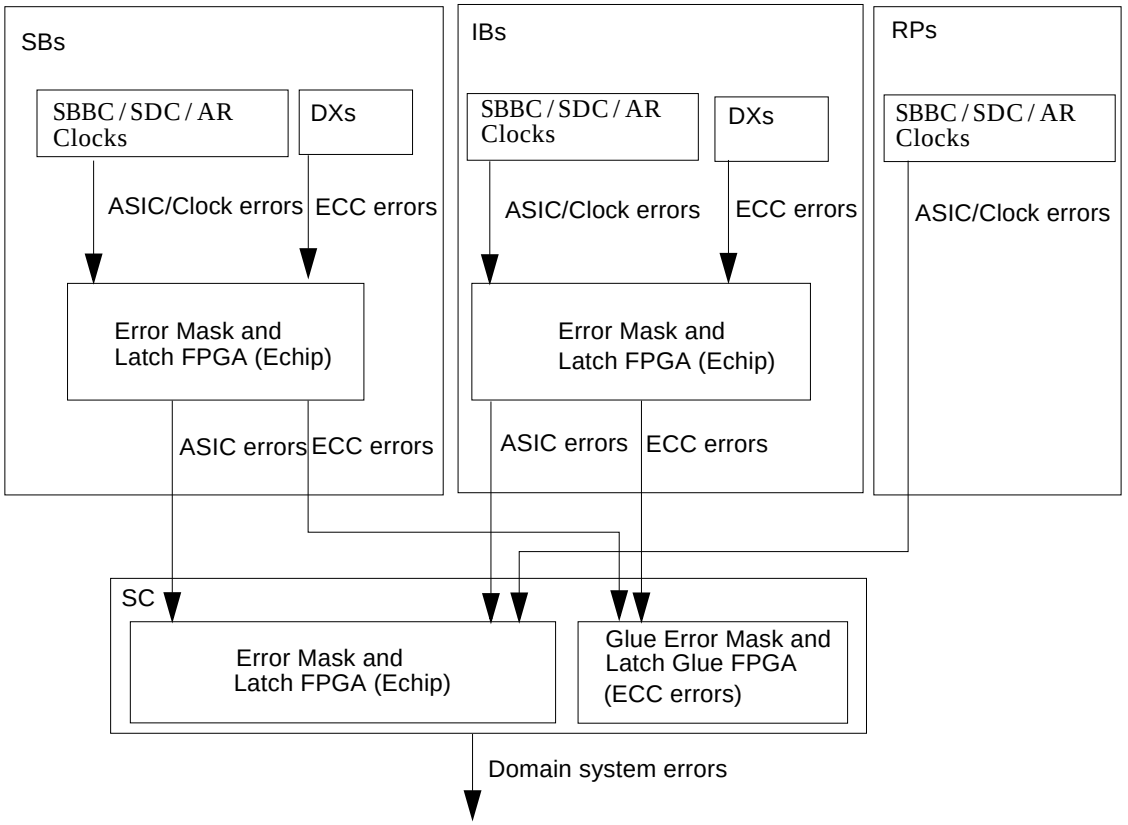


FIGURE 4-1 FPGA / Glue FPGA Chips

An FPGA on the SC collects the ASIC errors from Repeater, SB and IB boards. A separate FPGA Glue chip on the SC collects ECC errors from the SB and IB boards. Errors from these systems are similar to a tree, with the SC at the trunk, boards as the main branches, ASICs as sub-branches, error registers as sub-sub-branches.

Following is a typical SC response to an error:

1. **The FPGA or FPGA Glue chip on the SC records which FRU signalled the first error. If multiple errors were detected at the same times, then you will see multiple FRU first error reports as in the following example:**

CODE EXAMPLE 4-2 Multiple First Errors

```
Jul 11 13:37:38 u3800sc0 Domain-A.SC: /N0/IB8 encountered the first error
Jul 11 13:37:38 u3800sc0 Domain-A.SC: /N0/SB0 encountered the first error
Jul 11 13:37:38 u3800sc0 Domain-A.SC: /N0/SB2 encountered the first error
Jul 11 13:37:38 u3800sc0 Domain-A.SC: /N0/IB6 encountered the first error
```

In this case, SB0, SB2, IB6 and IB8 all reported errors at the same time.

2. **The SC then probes all boards and determines which ASIC within the FRU was the first to report an error (except for ECC errors). For each FRU that reported an error, you will see at least one ASIC that reported the first error for that FRU. If multiple ASICs reported simultaneously, you will see multiple ASIC first error reports as in the following example:**

CODE EXAMPLE 4-3 Multiple ASIC Errors

```
Oct 18 00:22:56 noname Domain-A.SC: [ID 533344 local0.error] SdcAsic reported
first error on /N0/SB3
Oct 18 00:22:57 noname Domain-A.SC: [ID 829288 local0.error]
/partition0/domain0/SB3/ar0:
...

Oct 18 00:22:57 noname Domain-A.SC: [ID 114318 local0.error]
/partition0/domain0/SB3/sdc0:
...

Oct 18 00:22:57 noname Domain-A.SC: [ID 689539
local0.error]/partition0/domain0/SB3/bbcGroup0/sbbc0:
...
Oct 18 00:22:59 noname Domain-A.SC: [ID 560560 local0.error]
/partition0/domain0/SB3/bbcGroup1/sbbc1:
...
```

In this case, on SB3, the SDC ASIC was the first to report an error for this FRU. In addition, the AR, BBC0 and BBC1 ASICs also reported errors (but later).

3. **The SC will dump each error register for every ASIC reporting an error. Error registers that do not contain errors will not be displayed. Through inspection of the FE bit, you can tell which error register within the ASIC was first to report an error. Multiple registers can detect errors simultaneously (and would all have the FE bit set).**

4. **Inspect the lower 14 bits of a register to determine which set of errors occurred first. These errors are also decoded and printed by the SC. For example, in the following message, bit 15, is printed out as FE. The upper 14 bits are sometimes decoded and printed. These are the accumulated errors that occurred after the first set captured in the lower 14 bits. For example:**

CODE EXAMPLE 4-4 Multiple SafariPortErrors/L2CheckError

```
Aug 20 19:30:59 appliance01 Domain-A.SC: [ID 538145 local0.error]
/partition0/domain0/SB2/sdc0:
>>> SafariPortError[0x200] : 0x00008002
               ParSglErr [01:01] : 0x1 ParitySingle error
                      FE [15:15] : 0x1
>>> SafariPortError1[0x210] : 0x00028002
            AccParSglErr [17:17] : 0x1
               ParSglErr [01:01] : 0x1 ParitySingle error
                      FE [15:15] : 0x1
>>> SafariPortError2[0x220] : 0x00008002
               ParSglErr [01:01] : 0x1 ParitySingle error
                      FE [15:15] : 0x1
>>> SafariPortError3[0x230] : 0x00008002
               ParSglErr [01:01] : 0x1 ParitySingle error
                      FE [15:15] : 0x1
    L2CheckError[0x6150] : 0x00e30022
        AccA1B1DiffErrDT [23:23] : 0x1
          A0A1DiffErrDT [01:01] : 0x1 L2 ports A0 and A1 mismatched (DTransID)
          A0B0DiffErrDT [05:05] : 0x1 L2 ports A0 and B0 mismatched (DTransID)
        AccA0A1DiffErrDT [17:17] : 0x1
        AccA1B1DiffErrTT [22:22] : 0x1
        AccA0B0DiffErrDT [21:21] : 0x1
 AccA0A1DiffErrTT [16:16] : 0x1
```

In this case (SB2, SDC ASIC), all four SafariPortError registers have the FE bit [15] set while the L2CheckError register does not. This indicates that the four Safari Port Errors happened first, before the L2CheckError.

Note that both the ParitySingle and AccParSglErr error bits are set for SafariPort 1. This indicates that a Parity Single error was the first error and was followed by one or more Parity Single error(s).

4.2 Console Bus Errors

The console bus is used by the SC to access system registers without having to rely on the integrity of the primary data and address busses. Only the SC can be master on the console bus. Each SC (and only the SC) has a CBH ASIC which acts as a hub for the console bus. The fourteen SC CBH slave ports are connected to an SDC ASIC on each of the SB, IB and RP boards. Each board's SDC ASIC, in turn, fans out the console bus internally depending upon the board type:

- SB boards - connects console bus to the AR and the two SBBC ASICS.
- IB boards - connects console bus to the AR and SBBC ASICs
- RP boards - connects the console bus to the AR ASIC only.

See Section A.5, "Console Bus Interconnect View" on page A-13 of the *Sun Fire 6800/4810/4800/3800 Systems Troubleshooting Manual* for more information.

Console bus errors are not reported by their own ASIC, but can be reported by any of the following ASICS:

- CBH: SC only
- SDC/AR: on SB, IB or RP boards
- SBBC: on SB, IB or RP boards

Consequently, `SYSTEM ERROR`s reported by the CBH, SDC, AR, and SBBC ASICs that also have the identifier, `ConsolePortError,` are likely Console Bus Errors. On SDC ASIC errors, the number after `ConsolePortError` is the repeater port number, no number means port zero.

TABLE 4-1 maps SDC ports to ASIC/boards:

TABLE 4-1 SDC Port Mapping to Boards

SDC Port	ASIC/Board
0 (master)	SBBC0/SB and IB boards only
1 (master)	SBBC1/SB and IB boards only
2 (master)	AR/all boards
3 (slave)	SC0
4 (slave)	SC1

CBH Port # mappings for SC:

0-14 Ports:

- 6 ports for SB boards,
- 4 ports for IB boards,
- 4 ports for repeater boards
- 1 port for SC (slave port 14)

Next, you need to interpret this register.

The format for the Console Bus Error Register (for all four ASICs) are nearly identical:

CBH: 9.3.10 Common error format (15 ports)

AR: 4.3.16 AR Console Port Error Register (one port)

SDC: 5.3.16S DC Console Port Error Register (5 ports, 0,1,2 = master, 3,4 slave)

SBBC: 3.3.2.5 SBBC Console Port Error Register (one port)

TABLE 4-2 ASIC Console Bus Error Register Format

Errror bit	ASIC	Description
CM_EACK	SBBC	Console master port got an error ack
CM_PERR	CBH, SDC, SBBC	Console master port data parity error
CM_PRER	CBH, SDC, SBBC	Console master port protocol error
CS_TO	CBH, SBBC	Console slave timeout
CS_ATO	CBH	Console slave arbitration timeout
CS_ILL	CBH, AR, SDC, SBBC	onsole slave detected an illegal access
CS_DPER	CBH, AR, SDC, SBBC	Console slave pairity eror on domain field during address phase
CS_APER	CBH, AR, SDC, SBBC	Console slave pairity error on Address field during address phase
CS_PERR	CBH, AR, SDC, SBBC	Console slave data parity error on write cycle
CS_PRER	CBH, AR, SDC, SBBC	Console slave protocol error

TODO: Define FRU list for each error bit & Reporting ASIC.

Notes: The console bus is a comm channel between the SC and a target FRU (going through the centerplane).

Can we identify FRUs for errors on one side of the SDC (e.g. AR and SBBC errors)?

4.2.1 Example of SB board AR `ConsolePortError/CsILL`

The following example is from an internal source: Bug 4485395. See also EMEA Radiance Case #36421128

CODE EXAMPLE 4-5 AR `ConsolePortError` on SB Board

```
Jul 31 15:48:14 sp4-sc0 Domain-A.SC: ErrorMonitor: Domain A has a SYSTEM ERROR
Jul 31 15:48:14 sp4-sc0 Domain-A.SC: /N0/SB2 encountered the first error
Jul 31 15:48:15 sp4-sc0 Domain-A.SC: ArAsic reported first error on /N0/SB2
Jul 31 15:48:15 sp4-sc0 Domain-A.SC: /partition0/domain0/SB2/ar0:
>>> ConsolePortError[0x400] : 0x00008010
 CsILL [04:04] : 0x1 Console slave port detected an illegal access
 FE [15:15] : 0x1

Jul 31 15:48:15 sp4-sc0 Domain-A.SC: Domain A is currently paused due to an
error.  This domain will be powered off and on via "setkeyswitch " to recover
Jul 31 15:48:15 sp4-sc0 Domain-A.SC: Keyswitch by
Thread[DomainMonitor.A,5,main] has begun:
on to offorrected Memory Error oPowering boards off ..
```

4.2.1.1 Analysis

This console port error is being reported by the AR ASIC on SB2. The component at fault will depend on error bits above.

4.2.1.2 Options for Recovery

- Setkeyswitch off and then on
- Reboot Sc
- XIR SC
- Power the cabinet off and on

4.2.1.3 Suspect FRUs

- SB2
- Centerplane

4.2.2 Example of SB Board AR and SBBC ConsolePortErrors and SDC SafariPortError

The following example is from a Sun Fire 6800 system and the source is: AMER Radiance Case 62721165.

CODE EXAMPLE 4-6 SB Board AR and SBBC ConsolePortError and SDC SafariPortError

```
noname:A> showlogs
Jul 11 11:30:29 noname Domain-A.POST: [ID 449568 local0.error] {/N0/SB4/P3}
Memory DIMM J16501 failed
Oct 18 00:22:56 noname Domain-A.SC: [ID 974093 local0.crit] Domain A has a SYSTEM
ERROR
Oct 18 00:22:56 noname Domain-A.SC: [ID 173191 local0.error] /N0/SB3 encountered
the first error
Oct 18 00:22:56 noname Domain-A.SC: [ID 533344 local0.error] SdcAsic reported
first error on /N0/SB3
Oct 18 00:22:57 noname Domain-A.SC: [ID 829288 local0.error]
/partition0/domain0/SB3/ar0:
>>> ConsolePortError[0x400] : 0x00018001
                  CsPREr [00:00] : 0x1 Console slave protocol error
               AccCsPREr [16:16] : 0x1
                      FE [15:15] : 0x1

Oct 18 00:22:57 noname Domain-A.SC: [ID 114318 local0.error]
/partition0/domain0/SB3/sdc0:
>>> SafariPortError1[0x210] : 0x00018001
                       FE [15:15] : 0x1
            AccParBidiErr [16:16] : 0x1
               ParBidiErr [00:00] : 0x1 ParityBidi error
    SafariPortError2[0x220] : 0x00000001
               ParBidiErr [00:00] : 0x1 ParityBidi error
    ConsolePortError[0x400] : 0x00010004
                AccCsPREr [16:16] : 0x1
                  CsAPEr [02:02] : 0x1 Console slave parity error on the Address
field during an address phase
.
.
.
.
.
See CODE EXAMPLE 4-7 for continuation.
```

CODE EXAMPLE 4-7 SB Board AR and SBBC Console Port Error / SDC Safari Port Error (Continuation from CODE EXAMPLE 4-6)

```
Oct 18 00:22:57 noname Domain-A.SC: [ID 689539 local0.error]
/partition0/domain0/SB3/bbcGroup0/sbbc0:
    ErrorStatus[0x80] : 0x80000000
                  ErrSum [31:31] : 0x1
>>> ConsolePortError[0x400] : 0x0001800c
                CsAper [02:02] : 0x1 Console slave parity error on the Address
field during an address phase
               AccCsPrer [16:16] : 0x1
                     FE [15:15] : 0x1
                  CsDper [03:03] : 0x1 Console slave parity error on domain
field during address phase

Oct 18 00:22:59 noname Domain-A.SC: [ID 560560 local0.error]
/partition0/domain0/SB3/bbcGroup1/sbbc1:
    ErrorStatus[0x80] : 0x80000000
                  ErrSum [31:31] : 0x1
>>> ConsolePortError[0x400] : 0x0001800c
                CsAper [02:02] : 0x1 Console slave parity error on the Address
field during an address phase
               AccCsPrer [16:16] : 0x1
                     FE [15:15] : 0x1
                  CsDper [03:03] : 0x1 Console slave parity error on domain
field during address phase

Oct 18 00:23:00 noname Domain-A.SC: [ID 253130 local0.crit] Domain A is
currently paused due to an error.  This domain must be turned off via
"setkeyswitch off" to recover
```

4.2.2.1 Suspect FRUs

- SB4, J16501
- SB3

4.2.3 Example of SDC ConsolePortError, SBBC ErrorStatus/SafErr, and DCDS Cheetah1ErrorStatus/C1DPerr Errors On an SB Board

The source for the following example is AMER Radiance Case #: 6274034.

CODE EXAMPLE 4-8 SB Board SDC ConsolePortError

```
Nov 14 21:49:08 bizsimdb-sc0 Domain-A.SC: POST detected paused domain, aborting
test.
Nov 14 21:49:08 bizsimdb-sc0 Domain-A.SC: Domain A has a SYSTEM ERROR
Nov 14 21:49:09 bizsimdb-sc0 Domain-A.SC: /N0/SB4 encountered the first error
Nov 14 21:49:10 bizsimdb-sc0 Domain-A.SC: RepeaterSbbcAsic reported first error
on /N0/SB4
Nov 14 21:49:10 bizsimdb-sc0 Domain-A.SC:/partition0/domain0/SB4/sdc0:
>>>ConsolePortError3[0x430] : 0x02008200
                 CmPErr [09:09] : 0x1 Console master port data parity error on
read data
               AccCmPErr [25:25] : 0x1
                      FE [15:15] : 0x1

Nov 14 21:49:10 bizsimdb-sc0 Domain-
A.SC:/partition0/domain0/SB4/bbcGroup1/sbbc1:
>>>ErrorStatus[0x80] : 0x800a8200
                AccBBIll [19:18] : 0x2
                      FE [15:15] : 0x1
                  ErrSum [31:31] : 0x1
                  SafErr [09:08] : 0x2 Fireplane device asserted an error
                AccBBErr [17:16] : 0x2

Nov 14 21:49:11 bizsimdb-sc0 Domain-
A.SC:/partition0/domain0/SB4/bbcGroup1/cpuCD/dcds7:
    Cheetah1ErrorStatus[0x61] : 0x00000200
 C1DPerr [09:09] : 0x1 Cheetah 1 data parity
```

4.2.3.1 Analysis

Isn't the first error between the SDC and the SC (SB4, SC0 and centerplane).... Example of an error that includes the SC!

RFE: Report both FRUs...

4.2.3.2 Suspect FRUs:

- SB4 since there is a parity error reported by a DCDS
- SC0
- Centerplane

4.3 Safari Port Errors

Safari Port Errors can be identified by the string `SafariPortError` in the output. The port number directly follows the `SafariPortError` string. If no number is printed, it is assumed to be port 0. This error can occur on SB, IB and repeater boards.

For the AR ASIC on a system or I/O board, ports 0 through 5 connect to internal devices on the board (CPUs or Schizos). Also, ports 6 through 9 are used to connect to L2 repeater boards (and thus cross FRU boundaries).

For the AR ASIC on the repeater boards, all ports (0 through 9) connect to either SB or IB boards (and thus cross FRU boundaries).

To determine which ports on an SB or IB board connect to which port on a specific repeater board (or vis-versa), refer to the table 3-14 (6800) or 3-15 (3800,4800,4810) in the troubleshooting guide. This will help you narrow down the list of FRUs.

Per the SAPR, there are only 3 unique error bits (per port):

QUNF_ERRQueue underflow

QOVF_ERRQueue overflow

ADR_PERRAddress parity

For failures on an SB or IB board where the port number is 0 through 5, the FRU is the reporting board. If the port number is 6 through 9 the failure could be the board itself, one of the repeater boards (which one?), or the centerplane.

For failures on a repeater board, the failure could be with the board itself, the slot associated with the port, or the centerplane.

4.3.1 Example of SB AR SafariPortError/AdrPErr Errors

The following example is from a single segmented SunFire 6800 system with SB4 in Domain A. The source is AMER Radiance Case #62603519.

CODE EXAMPLE 4-9 SB SafariPortError

```
it-sun99-sc Domain-A.SC: Domain A has a SYSTEM ERROR
Aug 03 06:12:37 it-sun99-sc Domain-A.SC: /N0/SB4 encountered the first error
Aug 03 06:12:37 it-sun99-sc Domain-A.SC: ArAsic reported first error on /N0/SB4
Aug 03 06:12:37 it-sun99-sc Domain-A.SC:
/partition0/domain0/SB4/ar0:
>>> SafariPortError3[0x230] : 0x00008001
                 AdrPErr [00:00] : 0x1 Address parity error
                      FE [15:15] : 0x1
Aug 03 06:12:37 it-sun99-sc Domain-A.SC: Domain A is currently paused due to an
error.  This domain must be turned off via "setkeyswitch off" to recover
```

4.3.1.1 Analysis

This is an address parity error against Safari Port number 3. Since ports 0-5 view inward onboard devices, the faulty FRU is the reporting board (SB4).

Note: lots of examples of address parity errors,, mostly during post, but not very clean. Appears as though the error might not always be fatal during POST.

4.3.1.2 Suspect FRU

- SB4

4.3.2 Transaction Count Overflow / Underflow Errors

You can identify AR transaction count underflow / overflow errors by the string `TransactionCountOVFError` or `TransactionCountUNFError`?. Like L2 Check Errors, these are only valid for SB or IB boards only.

Per the SAPR, there are only two unique errors:

- SLOT_OVFOverflow by slot number (bits 0-9)
- SLOT_UNFUnderflow by slot number (bits 0-9)

For each error type (underflow / overflow), there is a 10 bit field where each bit in the register indicates an error against that slot. You can decode the bit position to determine the slot.

These errors can be caused by either an internal ASIC error (FRU is the reporting board) or an L1 / L2 link failure. In the later case the failed FRU may be the reporting board, one of the other SB s (indicated by the slot number) or the centerplane.

TODO: In all of the above cases, when there are multiple possibilities, what are the most likely causes? GIVE guidance to the field so that best guess is made.

4.3.2.1 Example of IB `SafariPortErrors/L2 CheckError/TransactionCountOVFError` Errors

The following example is from a single segmented Sun Fire 6800 system with SB2 and IB6 in Domain A.

CODE EXAMPLE 4-10 IB SafariPortErrors / L2CheckError / TransactionCountOVFError

```
appliance01:A> showlogs
Aug 20 19:30:59 appliance01 Domain-A.SC: [ID 538145 local0.error]
/partition0/domain0/SB2/sdc0:
>>> SafariPortError[0x200] : 0x00008002
               ParSglErr [01:01] : 0x1 ParitySingle error
                      FE [15:15] : 0x1
>>> SafariPortError1[0x210] : 0x00028002
            AccParSglErr [17:17] : 0x1
               ParSglErr [01:01] : 0x1 ParitySingle error
                      FE [15:15] : 0x1
>>> SafariPortError2[0x220] : 0x00008002
               ParSglErr [01:01] : 0x1 ParitySingle error
                      FE [15:15] : 0x1
>>> SafariPortError3[0x230] : 0x00008002
               ParSglErr [01:01] : 0x1 ParitySingle error
                      FE [15:15] : 0x1
    L2CheckError[0x6150] : 0x00e30022
        AccA1B1DiffErrDT [23:23] : 0x1
          A0A1DiffErrDT [01:01] : 0x1 L2 ports A0 and A1 mismatched (DTransID)
          A0B0DiffErrDT [05:05] : 0x1 L2 ports A0 and B0 mismatched (DTransID)
        AccA0A1DiffErrDT [17:17] : 0x1
        AccA1B1DiffErrTT [22:22] : 0x1
        AccA0B0DiffErrDT [21:21] : 0x1
        AccA0A1DiffErrTT [16:16] : 0x1

Aug 20 19:31:04 appliance01 Domain-A.SC: [ID 974394
local0.error]/partition0/domain0/SB2/bbcGroup0/sbbc0:
>>> ErrorStatus[0x80] : 0x80008100
                      FE [15:15] : 0x1
                  ErrSum [31:31] : 0x1
                  SafErr [09:08] : 0x1 Fireplane device asserted an error

Aug 20 19:31:19 appliance01 Domain-A.SC: [ID 960880 local0.error]
/partition0/domain0/SB2/bbcGroup0/cpuAB/cpusafariagent0:
    AFAR (high)[0x531] : 0x00000010
            AFAR [42:32] [10:00] : 0x10
    AFSR (high)[0x551] : 0x00040000
                    IERR [18:18] : 0x1
>>> EMU A[0x501] : 0x00008000
                DROB_WER [15:15] : 0x1
See CODE EXAMPLE 4-11 for continuation.
```

CODE EXAMPLE 4-11 IB SafariPortErrors / `TransactionCountOVFError` (continued from CODE EXAMPLE 4-10)

```
Aug 20 19:31:24 appliance01 Domain-A.SC: [ID 727267 local0.error]
/partition0/domain0/SB2/bbcGroup1/sbbc1:
>>> ErrorStatus[0x80] : 0x82008100
                        FE [15:15] : 0x1
                    ErrSum [31:31] : 0x1

  SafErr [09:08] : 0x1 Fireplane device asserted an error
                AccSafErr [25:24] : 0x2

Aug 20 19:31:40 appliance01 Domain-A.SC: [ID 668986 local0.error]
/partition0/domain0/SB2/bbcGroup1/cpuCD/cpusafariagent0:
    AFAR (high)[0x531] : 0x00000014
            AFAR [42:32] [10:00] : 0x14
    AFAR (low)[0x541] : 0x04000000
    AFSR (high)[0x551] : 0x00080000
                    PERR [19:19] : 0x1
    EMU A[0x501] : 0x08000000
                     UDT [27:27] : 0x1

Aug 20 19:31:45 appliance01 Domain-A.SC: [ID 531593 local0.error] ArAsic
reported first error on /N0/IB6
Aug 20 19:31:50 appliance01 Domain-A.SC: [ID 347375 local0.error]
/partition0/domain0/IB6/ar0:
>>> TransactionCountOVFError[0x6100] : 0x02808200
               SlotOVF [09:00] : 0x200 Overflow for slot indicated by bit number
              AccSlotOVF [25:16] : 0x280
                      FE [15:15] : 0x1

Aug 20 19:31:55 appliance01 Domain-A.SC: [ID 347675 local0.error]
/partition0/domain0/IB6/sdc0:
>>> SafariPortError[0x200] : 0x00028002
             AccParSglErr [17:17] : 0x1
                ParSglErr [01:01] : 0x1 ParitySingle error
                       FE [15:15] : 0x1
>>> SafariPortError2[0x220] : 0x00028002
             AccParSglErr [17:17] : 0x1
                ParSglErr [01:01] : 0x1 ParitySingle error
                       FE [15:15] : 0x1
See  CODE EXAMPLE 4-12 for continuation.
```

CODE EXAMPLE 4-12 IB SafariPortErrors/`TransactionCountOVFError` (continued from CODE EXAMPLE 4-11)

```
L2CheckError[0x6150] : 0x00a20022
        AccA1B1DiffErrDT [23:23] : 0x1
          A0A1DiffErrDT [01:01] : 0x1 L2 ports A0 and A1 mismatched (DTransID)
          A0B0DiffErrDT [05:05] : 0x1 L2 ports A0 and B0 mismatched (DTransID)
        AccA0A1DiffErrDT [17:17] : 0x1
        AccA0B0DiffErrDT [21:21] : 0x1

Aug 20 19:32:00 appliance01 Domain-A.SC: [ID 744397 local0.error]
/partition0/domain0/IB6/dx1:
>>> Safari Port Error Status 6[0x25] : 0x00008004
               ScU3IFPar [02:02] : 0x1 Fireplane u3 input FIFO parity error
(ports 0 and 2 only)/Input parity error (other ports)
              FirstError [15:15] : 0x1

Aug 20 19:32:50 appliance01 Domain-A.SC: [ID 253130 local0.crit] Domain A is
currently paused due to an error.  This domain must be turned off via
"setkeyswitch off" to recover
```

Analysis

In cases where there are lots of errors, try to look for the error that most clearly indicates a failed FRU (or the most serious). In this case, we have SB2 reporting an IERR (internal error) for CPU A. Internal errors are pretty bad and can only be caused by a CPU. This error could have caused other errors, so the best course of action is to address this and see if any further errors occur (do one thing at a time).

Other errors include:

- SB2: PERR (protocol error) for CPU CFRU list unknown
- SB2: Parity errors for SDC ports 0-3 (internal paths to CPU)FRU list SB2
- SB2: L2 check errorsFRU list SB2, RP0-3, centerplane
- IB6: SDC parity error ports 0 & 2 (internal)FRU list IB6
- IB6: DX Parity error between IB6 & FRU list IB6,
- IB6: L2 check errors

A suspect FRU list would consist of IB6, RP0 through 3, or the centerplane.

Analyzing the overflow error. We see that IB6 is reporting an error. Decoding the bit fields (0x200 and 0x280... not sure why they are different) maps to IB7 and IB9 (or SB0 and SB2, depending upon direction). The latter might make more sense as we also have errors against SB2.

However, we have further information (see following example) from the system after the operator apparently tried to power the domain off:

CODE EXAMPLE 4-13 Domain Powered Off?

```
This will abruptly terminate Solaris in domain A.

Do you want to continue? [no] y

<<<...Bunch of noise deleted...>>>

Aug 21 15:25:50 appliance01 Domain-A.SC: Excluded unusable, failed or
disabled board: /N0/SB1

Aug 21 15:25:55 appliance01 Domain-A.SC: Excluded unusable, failed or
disabled board: /N0/SB3

Aug 21 15:26:01 appliance01 Domain-A.SC: Excluded unusable, failed or
disabled board: /N0/SB5

Aug 21 15:26:06 appliance01 Domain-A.SC: Excluded unusable, failed or
disabled board: /N0/IB7

Aug 21 15:26:11 appliance01 Domain-A.SC: Excluded unusable, failed or
disabled board: /N0/IB9
<<<...Bunch of noise deleted...>
```

Analysis

As you can see from the above example the system is having problems with one whole half of the system (all of the odd boards). This maps to Power Grid #1. Likely cause is that we lost power.

4.4 Remaining AR ASIC Errors (Excluding ConsoleBusErrors)

An AR ASIC is used on the SB and IB boards (L1 level AR) and the Repeater board (L2 level AR). This section excludes console bus errors reported by the AR (covered in a previous section) and lock step errors (not implemented).

There are three remaining categories of errors:

- L2 Check Errors, see Serengeti APR 4.3.12
- Safari Port Errors, see Serengeti APR 4.3.14
- Transaction Count Overflow / Underflow Errors, see Serengeti APR4.3.20, 4.3.21

4.4.1 L2 Check Errors

You can identify L2 Check Errors by the string `L2CheckError` in the output. These error types are only reported on SB or IB boards (not repeater boards).

Per the SAPR, there are only 4 unique error bits:

CMDV_SYNC_ERR[9:6]

INC_SYNC_ERR[9:6]

PREQ_SYNC_ERR[9:6]

DIFF_ERR

Most of these appear to be L1-L2 problems (note the port number 6 through 9 which indicates external).

Q: What about DIFF_ERR?

Per the troubleshooting manual, these errors can all be the fault of one of the following FRUs (not identifiable):

- SB or IB board reporting the error
- Either L2 repeater board (can we isolate one?)
- The centerplane

Note: If the failure is the fault of the L2 repeater board, multiple failures should report against the same slot. If the failure is the fault of the system board, the failures should follow the board. What about centerplane?

4.4.1.1 Example of IB Board L2CheckError/AccIncSyncErr/INCSyncErr

The following example is from a Sun Fire 4810 System and the source is Bug 4365883

CODE EXAMPLE 4-14 IB Board L2CheckError/INCSyncError

```
Oct 20 10:24:57 qame-sc0 Domain-B.SC: Domain B has a SYSTEM ERROR
Oct 20 10:24:57 qame-sc0 Domain-B.SC: /N0/IB6 encountered the first error
Oct 20 10:24:57 qame-sc0 Domain-B.SC: ArAsic reported first error on /N0/IB6
Oct 20 10:24:57 qame-sc0 Domain-B.SC: /partition0/domain1/IB6/ar0:
>>> L2CheckError[0x6150] : 0x01808180
 AccIncSyncErr [24:21] : 0xc accumulated incoming mismatch
 FE [15:15] : 0x1
 INCSyncErr [08:05] : 0xc Ports [9:6] incoming mismatched against internal
expected incoming

Oct 20 10:24:57 qame-sc0 Domain-B.SC: This domain is currently paused due to an
error.
This domain must be turned off via "setkeyswitch off" to recover
```

Analysis

This is an L2 Check Error reported by IB6. The error IncSyncErr doesn't matter. As errors of this type can have multiple causes, the possible FRUs are listed in the following section.

FRUs:

- IB6
- Centerplane
- Either repeater board

The following example is from a single segmented Sun Fire 4800 system with IB8 in Domain A. The source is AMER Radiance Case #62676166.

CODE EXAMPLE 4-15 IB Board L2CheckError/CMDVSyncError

```
Sep 23 02:44:53 lab-sf-con Domain-C.SC: Domain C has a SYSTEM ERROR
Sep 23 02:44:53 lab-sf-con Domain-C.SC: RP2 encountered the first error
Sep 23 02:44:53 lab-sf-con Domain-C.SC: ArAsic reported first error on /N0/IB8
Sep 23 02:44:54 lab-sf-con Domain-C.SC: /partition1/domain0/IB8/ar0:
>>> L2CheckError[0x6150] : 0x00008606
             CMDVSyncErr [12:09] : 0x3 Ports [9:6] command valid mismatched
against internal expected command valid
             PreqSyncErr [04:01] : 0x3 Ports [9:6] prereq mismatched against
internal expected prereq
                      FE [15:15] : 0x1

Sep 23 02:44:54 lab-sf-con Domain-C.SC:
/partition1/domain0/IB8/bbcGroup0/sbbc0:
>>> ErrorStatus[0x80] : 0x80008100

    FE [15:15] : 0x1
                 ErrSum [31:31] : 0x1
                 SafErr [09:08] : 0x1 Fireplane device asserted an error

Sep 23 02:44:55 lab-sf-con Domain-C.SC: Domain C is currently paused due to an
error.This domain must be turned off via "setkeyswitch off" to recover
```

4.4.1.2 Analysis

This is an `L2CheckError` reported by IB8. The type of error is different since there is an additional `CMDVSyncError` error reported. As errors of this type can have multiple causes, the possible FRUs are listed in the section that folows.

Note – Since this error is reported by Schizo #0, it can't be blacklisted to try and isolate the error due to it's connection to the SBBC.

4.4.1.3 Suspect FRUs

- SB0
- IB8
- Either repeater board

4.4.2 Example of L2CheckErrors/CMDXVSyncErrors on an IB Board

The folowing example is from a single segmented Sun Fire 3800 system with SB0, SB2, IB6 and IB8 in Domain A.

CODE EXAMPLE 4-16 IB Board Multiple L2CheckErrors/CMDXVSyncErrors

```
Jul 11 13:37:37 u3800sc0 Domain-A.SC: Domain A has a SYSTEM ERROR
Jul 11 13:37:38 u3800sc0 Domain-A.SC: /N0/IB8 encountered the first error
Jul 11 13:37:38 u3800sc0 Domain-A.SC: /N0/SB0 encountered the first error
Jul 11 13:37:38 u3800sc0 Domain-A.SC: /N0/SB2 encountered the first error
Jul 11 13:37:38 u3800sc0 Domain-A.SC: /N0/IB6 encountered the first error
Jul 11 13:37:38 u3800sc0 Domain-A.SC: ArAsic reported first error on /N0/IB8
Jul 11 13:37:38 u3800sc0 Domain-A.SC:/partition0/domain0/IB8/ar0:
>>> L2CheckError[0x6150] : 0x00019818
             CMDVSyncErr [12:09] : 0xc Ports [9:6] command valid mismatched
against internal expected command valid
             PreqSyncErr [04:01] : 0xc Ports [9:6] prereq mismatched against
internal expected prereq
                      FE [15:15] : 0x1
              AccDiffErr
 [16:16] : 0x1

Jul 11 13:37:38 u3800sc0 Domain-A.SC: ArAsic reported first error on /N0/SB0
Jul 11 13:37:38 u3800sc0 Domain-A.SC: /partition0/domain0/SB0/ar0:
>>> L2CheckError[0x6150] : 0x00008180
                      FE [15:15] : 0x1
             INCSyncErr [08:05] : 0xc Ports [9:6] incoming mismatched against
internal expected incoming

Jul 11 13:37:38 u3800sc0 Domain-A.SC: ArAsic reported first error on /N0/SB2
Jul 11 13:37:38 u3800sc0 Domain-A.SC: /partition0/domain0/SB2/ar0:
>>> L2CheckError[0x6150] : 0x00019818
             CMDVSyncErr [12:09] : 0xc Ports [9:6] command valid mismatched
against internal expected command valid
             PreqSyncErr [04:01] : 0xc Ports [9:6] prereq mismatched against
internal expected prereq
                      FE [15:15] : 0x1
              AccDiffErr [16:16] : 0x1
              AccDiffErr [16:16] : 0x1
.
.
.
See CODE EXAMPLE 4-17 for continuation.
```

CODE EXAMPLE 4-17 IB Board Multiple L2CheckErrors/CMDXVSyncErrors (continued)

```
Jul 11 13:37:38 u3800sc0 Domain-A.SC: ArAsic reported first error on /N0/IB6
Jul 11 13:37:38 u3800sc0 Domain-A.SC: /partition0/domain0/IB6/ar0:
>>> L2CheckError[0x6150] : 0x0001
9818
             CMDVSyncErr [12:09] : 0xc Ports [9:6] command valid mismatched
against internal expected command valid
             PreqSyncErr [04:01] : 0xc Ports [9:6] prereq mismatched against
internal expected prereq
                      FE [15:15] : 0x1
              AccDiffErr [16:16] : 0x1

Jul 11 13:37:38 u3800sc0 Domain-A.SC: Domain A is currently paused due to an
error.  This domain must be turned off via "setkeyswitch off" to recover
```

4.4.2.1 Analysis

This has L2 Check errors against four boards. Three of the boards have the same error, while a fourth is unique. You can use two different techniques.

Assume that all four boards are not bad, then you would suspect the two repeater boards (RP0 or RP2). If the machine is not segmented (and we can't tell since this is domain A), you might consider segmenting the machine (would this make sense on a 3800 using both IB boards... we can't have another domain?).

Then, when the system fails, you can swap the two repeater boards (RP0 & RP2) to see if the failure goes to the other domain (but we can't have another domain on a 3800 using both IBs...).

Assume that odd board SB0 is the problem (has different error).

Assume the problem is slot specific or the centerplane.

Note that AccDiffErr is set, while DiffErr is not. What about Diff error... Conclusion????

Bizzare note: (EMEA Case #36414737) has the exact same error but is a different system...

4.4.2.2 Suspect FRUs:

- SB0
- IB8,IB6
- Either repeater board

- Centerplane

4.5 Remaining SDC ASIC Errors (Excluding ConsoleBusErrors)

The SDC ASIC is used on the SB (L1 CPU mode), IB (L1 IO mode) and Repeater boards (L2 mode). The SDC has two major functions. First, it implements the control for the 2-level Safari data interconnect: second, it implements a Console Bus multiplexor.

This section excludes console bus errors reported by the SDC (covered in a previous section) and lock step errors (not implemented) and ECC error status (covered in a previous section).

There are two remaining categories of errors:

- Safari Port Errors, see Serengeti APR 5.3.11
- L2 Check Errors, see Serengeti APR 5.3.13

4.5.1 Safari Port Errors

Safari Port Errors are identified by the string `SafariPortError` in the output. The port number directly follows the `SafariPortError` string. If no number is printed, it is assumed to be port 0. In the case of an SB or IB board, ports 0-3 are connected to internal devices; ports 4-9 are connected to the L2 repeater boards.

To identify the specific repeater board that is connected to one of ports 4-9, refer to TABLE D-4 in the Sun Fire 6800/4810/4800/3800 Systems Troubleshooting Manual. In the case of an L2 repeater port, the port number of the repeater will match the physical slot of the SB or IB board. For example, as shown in TABLE D-4, repeater slot 0 is connected to SB0 and slot 6 is connected to IB6.

Per the SAPR, there are 10 unique error bits:

- ARB_ERR_DT
- ARB_ERR_TT
- ID_ERR_DT
- ID_ERR_TT
- QOVF_ERR_DT
- QOVF_ERR_TT
- PAR_L2_ERR_DT
- PAR_L2_ERR_TT
- PAR_SGL_ERR

- PAR_BIDI_ERR

Note: Don't quite understand the language of the Tshoot guide on recommendations.

4.5.1.1 Example of SB `SafariPortError/ParSglErr` on SDC Port 2

The following source is from internal Bug # 4392515.

CODE EXAMPLE 4-18 SB SafariPortError on SDC Port 2

```
/N0/SB0/P0:  CPU 19 clearing 00000004.d5555480 to 00000004.eaaaa9c0
/N0/SB0/P0:  CPU 0 clearing 00000004.eaaaa9c0 to 00000005.00000000
Nov 27 09:48:08 qamd1-sc0 Domain-A.SC: Domain A has a SYSTEM ERROR
Nov 27 09:48:08 qamd1-sc0 Domain-A.SC: /N0/SB2 encountered the first error
Nov 27 09:48:08 qamd1-sc0 Domain-A.SC: SdcAsic reported first error on /N0/SB2
Nov 27 09:48:08 qamd1-sc0 Domain-A.SC: /partition0/domain0/SB2/sdc0:
>>> SafariPortError2[0x220] : 0x00008002
               ParSglErr [01:01] : 0x1 ParitySingle error
                      FE [15:15] : 0x1

Nov 27 09:48:08 qamd1-sc0 Domain-A.SC: This domain is currently paused due to
an error.
This domain must be turned off via "setkeyswitch off" to recover.
```

Analysis

Refering to TABLE D-4 of the Sun Fire 6800 / 4810 / 4800 / 3800 Troubleshooting Manual, this is the fault of SB2.

Suspect FRU

- SB2

4.5.1.2 Repeater Board SafariPortError/ParL2ErrTT on SDC Port 0

The following source is from internal Bug 4382329

CODE EXAMPLE 4-19 Repeater board SafariPortError on SDC Port 0

```
ds1-sc0:SC> Jan 01 07:46:48 ds1-sc0 Chassis-Port.SC: Domain A has a SYSTEM ERROR
Jan 01 07:46:51 ds1-sc0 Chassis-Port.SC: This domain is still running because
error pause is not enabled for this domain
Jan 01 08:28:49 ds1-sc0 Chassis-Port.SC: Domain A has a SYSTEM ERROR
Jan 01 08:28:49 ds1-sc0 Chassis-Port.SC: /partition0/RP2/sdc0:
>>> SafariPortError[0x200] : 0x00008004
              ParL2ErrTT [02:02] : 0x1 L2 parity error for TTransID
                     FE [15:15] : 0x1

Jan 01 08:28:49 ds1-sc0 Chassis-Port.SC: This domain is currently paused due to
an error.
This domain must be turned off via "setkeyswitch off" to recover.
```

Analysis

This was reported by repeater board 2, on SDC port 0. Refering to the Sun Fire 6800/4810/4800/3800 Troubleshooting Manual and TABLE D-4, Repeater port 0 maps to SB0.

Suspect FRUs:

- RP2
- SB0
- Centerplane

4.5.2 L2 Check Errors

L2 Check Errors are identified by the string `L2CheckError` in the output. Per the SAPR, there are 8 unique error bits. However, since we have the following mappings, there are only two unique errors, with four combinations of repeater board pairs.:

- A0 = RP0
- A1 = RP1
- B0 = RP2
- B1 = RP3

- A1B1_DIFF_ERR_DTRP1 & RP3DtransID mismatch
- A1B1_DIFF_ERR_TTRP1 & RP3TtransID mismatch
- A0B0_DIFF_ERR_DTRP0 & RP2TtransID mismatch
- A0B0_DIFF_ERR_TTRP0 & RP2DtransID mismatch
- B0B1_DIFF_ERR_DTRP2 & RP3TtransID mismatch
- B0B1_DIFF_ERR_TTRP2 & RP3DtransID mismatch
- A0A1_DIFF_ERR_DTRP0 & RP1DtransID mismatch
- A0A1_DIFF_ERR_TTRP0 & RP1TtransID mismatch

Q: Do we use the same diagnostic technique for AR L2 Check errors? Can they only occur on system and I/O boards? See Code example 1-33 in troubleshooting guide.

4.5.2.1 Example of SB Board SDC `L2CheckError`

The source of the following is Internal Bug 4433403.

CODE EXAMPLE 4-20 SB Board SDC L2CheckError

```
Copying IO prom to Cpu dram
.
Apr 02 21:29:52 sc-widowmaker Domain-A.SC: POST detected paused domain, aborting
test.
Apr 02 21:29:52 sc-widowmaker Domain-A.SC: Domain A has a SYSTEM ERROR
Apr 02 21:29:52 sc-widowmaker Domain-A.SC: /N0/SB2 encountered the first error
Apr 02 21:29:52 sc-widowmaker Domain-A.SC: SdcAsic reported first error on
/N0/SB2
Apr 02 21:29:52 sc-widowmaker Domain-A.SC: /partition0/domain0/SB2/sdc0:
>>> L2CheckError[0x6150] : 0x00008020
          A0B0DiffErrDT [05:05] : 0x1 L2 ports A0 and B0 mismatched (DTransID)
                        FE [15:15] : 0x1

Apr 02 21:29:52 sc-widowmaker Domain-A.SC: Domain A is currently paused due to
an error.
This domain must be turned off via "setkeyswitch off" to recover.
```

Analysis

Using the same technique as the AR L2 Check errors, this is either the fault of SB2, one of the repeater boards (RP0 or RP2), or the centerplane.

- RP0 or RP2
- SB2
- Centerplane

4.6 Remaining DX ASIC Errors

The DX ASIC is the data crossbar used on the SB, IB and repeater boards. This section excludes lock step errors (not implemented), data parity errors, and ECC error status (see).

There are two remaining categories of errors:

- General Errors, see Serengeti APR 6.3.10
- Safari Port Errors, see Serengeti APR 6.3.13

4.6.1 General Errors

You can identify General Errors by the string `General Error Status` in the output. Per the SAPR, there are only 4 unique error bits (exluding lockstep):

- PTOUT Pause Timeout (4 pause lines)
- LOCKSTEP23
- LOCKSTEP01
- MOWNMAPB Multiple Owned/Mapped error for port B
- MOWNMAPA Same for port A
- CPERR Control parity error (SDC to DX)

For CPERR the FRU is probably the reporting board. Q: What about PTOUT and MOWN? What does port A/B Mean?

4.6.1.1 Example of SB Board DX `PTimeout` Error

The source for the following is Internal Bug 4392515.

CODE EXAMPLE 4-21 SB Board DX Pause Timeout

```
Mar 16 18:55:23 ug101 Domain-A.SC: [ID 604870 local0.warning] POST detected
paused domain, aborting test.
Mar 16 18:55:23 ug101 Domain-A.SC: [ID 974093 local0.crit] Domain A has a SYSTEM
ERROR
Mar 16 18:55:23 ug101 Domain-A.SC: [ID 627387 local0.error] /N0/SB0 encountered
the first error
Mar 16 18:55:23 ug101 Domain-A.SC: [ID 200480 local0.error] DxAsic reported
first error on /N0/SB0
Mar 16 18:55:23 ug101 Domain-A.SC: [ID 110091 local0.error]
/partition0/domain0/SB0/dx0:
>>> General Error Status[0x1e] : 0x04008400
              FirstError [15:15] : 0x1
                PTimeout [11:08] : 0x4 Pause timeout

Mar 16 18:55:23 ug101 Domain-A.SC: [ID 253130 local0.crit] Domain A is currently
paused due to an error.  This domain must be turned off via "setkeyswitch off"
to recover.
```

Analysis

Suspect FRUs:

- SB0
- If 6800 single segment: RP0, RP2
- If 6800 dual segment domain A or B: RP0
- If 6800 dual segment domain C or D: RP2
- If 4810/4800/3800 single segment: RP0, RP2
- If 4810/4800/3800 dual segment domain A: RP0
- If 4810/4800/3800 dual segment domain C: RP2

4.6.1.2 Example of DX General Errors

The following example is from a single segmented Sun Fire 6800 system with SB0, SB3, and SB4 in Domain A. The source is AMER Radiance Case #62746579.

CODE EXAMPLE 4-22 SYSTEM ERROR/ConsoleBus ERRORs/SafariPort Errors/General Errors

```
Nov 14 02:25:29 noname Chassis-Port.SC: Domain A has a SYSTEM ERROR
Pcf8591 write error ignored
Pcf8591 write error ignored
Pcf8591 write error ignored
Pcf8591 write error ignored
JtagController.tapWait: sun.serengeti.CommException: ConsoleBus ERROR:
errorCode=00000800 (CM_EACK) ack=ee
        CBH Port 14 (SSC0) error=00000040 (CS_TO)
        I/O request: IB6.sdc.b0 (12c000b0) offset=02c000b0 window=81000004 P=0
DD=1 space=4
        Error address: IB6.sdc.b0 (12c000b0)
JtagController.tapWait: sun.serengeti.CommException: ConsoleBus ERROR:
errorCode=00000800 (CM_EACK) ack=ee
        CBH Port 14 (SSC0) error=00000040 (CS_TO)
        I/O request: IB6.sdc.b0 (12c000b0) offset=02c000b0 window=81000004 P=0
DD=1 space=4
        Error address: IB6.sdc.b0 (12c000b0)
See  CODE EXAMPLE 4-23 for continuation.
```

CODE EXAMPLE 4-23 SYSTEM ERROR/ConsoleBus ERRORs/SafariPort Errors/General Errors (continued from CODE EXAMPLE 4-22)

```
Nov 14 02:25:36 noname Chassis-Port.SC: PCI I/O Board at /N0/IB6 Device poll
caused: sun.serengeti.HpuFailedException: VoltageA2D.getOutputVoltage:
sun.serengeti.I2cException: I2cComm.busyWait: busyWait() timeout waiting for
RRDY, status=0x304bc008,bus=12(/N0/IB6) ring=03 addr=4b

Nov 14 02:25:37 noname Chassis-Port.SC: Device will not be polled
Nov 14 02:25:37 noname Chassis-Port.SC: PCI I/O Board at /N0/IB6 Device poll
caused: sun.serengeti.HpuFailedException:
VoltageA2D.getOutputVoltage: sun.serengeti.I2cException: I2cComm.busyWait:
busyWait() timeout waiting for RRDY, status=0x304bc008,
bus=12(/N0/IB6) ring=03 addr=4b

Nov 14 02:25:37 noname Chassis-Port.SC: Device will not be polled
Nov 14 02:25:37 noname Chassis-Port.SC: PCI I/O Board at /N0/IB6 Device poll
caused: sun.serengeti.HpuFailedException:
ScVoltageA2D.getOutputVoltage: sun.serengeti.I2cException: I2cComm.busyWait:
busyWait() timeout waiting for RRDY, status=0x304bc008,
bus=12(/N0/IB6) ring=03 addr=4b

Nov 14 02:25:38 noname Chassis-Port.SC: Device will not be polled
Nov 14 02:25:38 noname Chassis-Port.SC: PCI I/O Board at /N0/IB6 Device poll
caused: sun.serengeti.HpuFailedException:
IoVoltageA2D.getOutputVoltage: sun.serengeti.I2c
Exception: I2cComm.busyWait: busyWait() timeout waiting for RRDY,
status=0x304bc008
bus=12(/N0/IB6) ring=03 addr=4b
Nov 14 02:25:38 noname Chassis-Port.SC: Device will not be polled
Nov 14 02:25:38 noname Chassis-Port.SC: PCI I/O Board at /N0/IB6 Device poll
caused: sun.serengeti.FailedHwException: Lm75 getTemp:
I2cComm.busyWait: busyWait() timeout waiting for RRDY, status=0x304cc008,
bus=12(/N0/IB6) ring=03 addr=4c
Nov 14 02:25:38 noname Chassis-Port.SC: Device will not be polled
Nov 14 02:25:38 noname Chassis-Port.SC: PCI I/O Board at /N0/IB6 Device poll
caused: sun.serengeti.FailedHwException: Lm75 getTemp:
I2cComm.busyWait: busyWait() timeout waiting for RRDY, status=0x304dc008,
bus=12(/N0/IB6) ring=03 addr=4d
See CODE EXAMPLE 4-24 for continuation.
```

CODE EXAMPLE 4-24 SYSTEM ERROR/ConsoleBus ERRORs/SafariPort Errors/General Errors (continued from CODE EXAMPLE 4-23)

```
Nov 14 02:25:38 noname Chassis-Port.SC: Device will not be polled
Nov 14 02:25:39 noname Chassis-Port.SC: PCI I/O Board at /N0/IB6 Device poll
caused: sun.serengeti.FailedHwException:
(SdcAsic)Asic.getTemp: ConsoleBus ERROR: errorCode=00000800 (CM_EACK) ack=ee


 CBH Port 14 (SSC0) error=00000040 (CS_TO)
        I/O request: IB6.sdc.10 (12c00010) offset=02c00010 window=81000004 P=0
DD=1 space=4
        Error address: IB6.sdc.10 (12c00010)
Nov 14 02:25:39 noname Chassis-Port.SC: Device will not be polled
Nov 14 02:25:39 noname Chassis-Port.SC: PCI I/O Board at /N0/IB6 Device poll
caused: sun.serengeti.FailedHwException:
(ArAsic)Asic.getTemp: ConsoleBus ERROR: errorCode=00000800 (CM_EACK) ack=ee

        CBH Port 14 (SSC0) error=00000040 (CS_TO)
        I/O request: IB6.ar.10 (12c80010) offset=02c80010 window=81000004 P=0
DD=1 space=4
        Error address: IB6.ar.10 (12c80010)
Nov 14 02:25:39 noname Chassis-Port.SC: Device will not be polled
Nov 14 02:25:39 noname Chassis-Port.SC: PCI I/O Board at /N0/IB6 Device poll
caused: sun.serengeti.FailedHwException:
/partition0/domain0/IB6/dx0: DxAsic.getTemp: sun.serengeti.jtag.JtagException:
JtagController.tapWait: sun.serengeti.CommException:
ConsoleBus ERROR: errorCode=00000800 (CM_EACK) ack
=ee

        CBH Port 14 (SSC0) error=00000040 (CS_TO)
        I/O request: IB6.sdc.b0 (12c000b0) offset=02c000b0 window=81000004 P=0
DD=1 space=4
        Error address: IB6.sdc.b0 (12c000b0)
Nov 14 02:25:40 noname Chassis-Port.SC: Device will not be polled
Nov 14 02:25:40 noname Chassis-Port.SC: PCI I/O Board at /N0/IB6 Device poll
caused: sun.serengeti.FailedHwException:
/partition0/domain0/IB6/dx1: DxAsic.getTemp: sun.serengeti.jtag.JtagException:
JtagController.tapWait: sun.serengeti.CommException:
ConsoleBus ERROR: errorCode=00000800 (CM_EACK) ack=ee
See CODE EXAMPLE 4-25 for continuation.
```

CODE EXAMPLE 4-25 SYSTEM ERROR/ConsoleBus ERRORs/SafariPort Errors/General Errors (continued from CODE EXAMPLE 4-24)

```
CBH Port 14 (SSC0) error=00000040 (CS_TO)
        I/O request: IB6.sdc.b0 (12c000b0) offset=02c000b0 window=81000004 P=0
DD=1 space=4
        Error address: IB6.sdc.b0 (12c000b0)
Nov 14 02:25:40 noname Chassis-Port.SC: Device will not be polled
Nov 14 02:25:41 noname Chassis-Port.SC: PCI I/O Board at /N0/IB6 Device poll
caused: sun.serengeti.FailedHwException:
(RepeaterSbbcAsic)Asic.getTemp: ConsoleBus ERROR: errorCod
e=00000800 (CM_EACK) ack=ee

        CBH Port 14 (SSC0) error=00000040 (CS_TO)
        I/O request: IB6.sbbc0.regs.10 (11800010) offset=01800010
window=81000004 P=0 DD=1 space=4
        Error address: IB6.sbbc0.regs.10 (11800010)
Nov 14 02:25:41 noname Chassis-Port.SC: Device will not be polled
Nov 14 02:25:41 noname Chassis-Port.SC: PCI I/O Board at /N0/IB6 Device poll
caused: sun.serengeti.FailedHwException:
Max1617Base.getTemp: sun.serengeti.I2cException: I2cComm.readCmd:
sun.serengeti.CommException: ConsoleBus ERROR:
errorCode=00000800 (CM_EACK) ack=ee

        CBH Port 14 (SSC0) error=00000040 (CS_TO)
        I/O request: IB6.sbbc0.regs.c0 (118000c0) offset=018000c0
window=81000004 P=0 DD=1 space=4
        Error address: IB6.sbbc0.regs.c0 (118000c0)
Nov 14 02:25:41 noname Chassis-Port.SC: Device will not be polled
Nov 14 02:25:42 noname Chassis-Port.SC: PCI I/O Board at /N0/IB6 Device poll
caused: sun.serengeti.FailedHwException:
Max1617Base.getTemp: sun.serengeti.I2cExcep
tion: I2cComm.readCmd: sun.serengeti.CommException: ConsoleBus ERROR:
errorCode=00000800 (CM_EACK) ack=ee

        CBH Port 14 (SSC0) error=00000040 (CS_TO)
        I/O request: IB6.sbbc0.regs.c0 (118000c0) offset=018000c0
window=81000004 P=0 DD=1 space=4
        Error address: IB6.sbbc0.regs.c0 (118000c0)
Nov 14 02:25:42 noname Chassis-Port.SC: Device will not be polled
Nov 14 02:25:42 noname Chassis-Port.SC: /partition0/RP2/ar0:
>>> SafariPortError6[0x260] : 0x00038001
                AccQOvf [17:17] : 0x1 accumulated queue overflow
                AdrPErr [00:00] : 0x1 Address parity error
                     FE [15:15] : 0x1
             AccAdrPErr [16:16] : 0x1 accumulated address parity errors
    SafariPortError7[0x270] : 0x00030001
                AccQOvf [17:17] : 0x1 accumulated queue overflo
See CODE EXAMPLE 4-26 for continuation.
```

CODE EXAMPLE 4-26 SYSTEM ERROR/ConsoleBus ERRORs/SafariPort Errors/General Errors (continued from CODE EXAMPLE 4-25)

```
 AdrPErr [00:00] : 0x1 Address parity error
             AccAdrPErr [16:16] : 0x1 accumulated address parity errors

Nov 14 02:25:43 noname Chassis-Port.SC: /partition0/RP2/sdc0:
>>> SafariPortError6[0x260] : 0x00cc8008
                       FE [15:15] : 0x1
          AccParL2ErrDT [19:19] : 0x1
             AccIDErrDT [23:23] : 0x1
             ParL2ErrDT [03:03] : 0x1 L2 parity error for DTransID
          AccParL2ErrTT [18:18] : 0x1
             AccIDErrTT [22:22] : 0x1

Nov 14 02:25:43 noname Chassis-Port.SC: /partition0/RP2/dx0:
    General Error Status[0x1e] : 0x00010001
                AccCPerr [16:16] : 0x1
                   CPerr [00:00] : 0x1 Control Parity Error
    Safari Port Error Status 6[0x25] : 0x00050004
            AccScU3IFPar [18:18] : 0x1
               ScU3IFPar [02:02] : 0x1 Fireplane u3 input FIFO parity error
(ports 0 and 2 only)/Input parity error (other ports)

                AccIFUf [16:16] : 0x1
>>> Safari Port Error Status 7[0x26] : 0x00058004
            AccScU3IFPar [18:18] : 0x1
               ScU3IFPar [02:02] : 0x1 Fireplane u3 input FIFO parity error
(ports 0 and 2 only)/Input parity error (other ports)

                AccIFUf [16:16] : 0x1
              FirstError [15:15] : 0x1

Nov 14 02:25:44 noname Chassis-Port.SC: /partition0/RP2/dx1:
    General Error Status[0x1e] : 0x00010001
                AccCPerr [16:16] : 0x1
                   CPerr [00:00] : 0x1 Control Parity Error
    Safari Port Error Status 6[0x25] : 0x00050004
            AccScU3IFPar [18:18] : 0x1
               ScU3IFPar [02:02] : 0x1 Fireplane u3 input FIFO parity error
(ports 0 and 2 only)/Input parity error (other ports)

                AccIFUf [16:16] : 0x1
>>> Safari Port Error Status 7[0x26] : 0x00058004
            AccScU3IFPar [18:18] : 0x1
               ScU3IFPar [02:02] : 0x1 Fireplane u3 input FIFO parity error
(ports 0 and 2 only)/Input parity error (other ports)
See CODE EXAMPLE 4-27 for continuation.
```

CODE EXAMPLE 4-27 ISYSTEM ERROR/ConsoleBus ERRORs/SafariPort Errors/General Errors (continued from CODE EXAMPLE 4-26)

```
 AccIFUf [16:16] : 0x1
              FirstError [15:15] : 0x1
Nov 14 02:25:45 noname Chassis-Port.SC: /partition0/RP0/ar0:
>>> SafariPortError6[0x260] : 0x00038001
                 AccQOvf [17:17] : 0x1 accumulated queue overflow
                 AdrPErr [00:00] : 0x1 Address parity error
                      FE [15:15] : 0x1
              AccAdrPErr [16:16] : 0x1 accumulated address parity errors
    SafariPortError7[0x270] : 0x00030001
                 AccQOvf [17:17] : 0x1 accumulated queue overflow
                 AdrPErr [00:00] : 0x1 Address parity error
              AccAdrPErr [16:16] : 0x1 accumulated address parity errors
Nov 14 02:25:45 noname Chassis-Port.SC: /partition0/RP0/sdc0:
>>> SafariPortError6[0x260] : 0x00cc8004
              ParL2ErrTT [02:02] : 0x1 L2 parity error for TTransID
                      FE [15:15] : 0x1
           AccParL2ErrDT [19:19] : 0x1
              AccIDErrDT [23:23] : 0x1
           AccParL2ErrTT [18:18] : 0x1
              AccIDErrTT [22:22] : 0x1
Nov 14 02:25:45 noname Chassis-Port.SC: /partition0/RP0/dx0:
    Safari Port Error Status 6[0x25] : 0x00040004
            AccScU3IFPar
 [18:18] : 0x1
               ScU3IFPar [02:02] : 0x1 Fireplane u3 input FIFO parity error
(ports 0 and 2 only)/Input parity error (other ports)
>>> Safari Port Error Status 7[0x26] : 0x00048004
            AccScU3IFPar [18:18] : 0x1
               ScU3IFPar [02:02] : 0x1 Fireplane u3 input FIFO parity error
(ports 0 and 2 only)/Input parity error (other ports)
              FirstError [15:15] : 0x1
Nov 14 02:25:46 noname Chassis-Port.SC: /partition0/RP0/dx1:
>>> Safari Port Error Status 6[0x25] : 0x00048004
            AccScU3IFPar [18:18] : 0x1
               ScU3IFPar [02:02] : 0x1 Fireplane u3 input FIFO parity error
(ports 0 and 2 only)/Input parity error (other ports)
              FirstError [15:15] : 0x1
    Safari Port Error Status 7[0x26] : 0x00040004
            AccScU3IFPar [18:18] : 0x1
               ScU3IFPar [02:02] : 0x1 Fireplane u3 input FIFO parity error
(ports 0 and 2 only)/Input parity error (other ports)
Nov 14 02:25:47 noname Chassis-Port.SC: Domain A is currently paused due to an
error.  This domain must be turned off via "setkeyswitch off" to recover
```

Analysis

The are two blocks of errors, the first are errors reported by the SC while trying to talk to IB6. Normally, you would suspect the SC, IB6 or the centerplane. The remaining errors (system error) all point to problems reported between RP0, RP2 and IB6. Combining these two seem to indicate that something is wrong with IB6 (e.g. It lost power or died).

Suspect FRU

- IB6

4.6.1.3 Example of Various Parity Errors

The following is taken from the loghost on a Sun Fire 3800 System. The source is AMER Radiance Case #62754759.

CODE EXAMPLE 4-28 System Error Followed by Various Parity Errors

```
Nov 19 23:20:43 [10.241.100.55.4.0] Domain-A.SC [ID 724995 local1.crit]
ErrorMonitor: Domain A has a SYSTEM ERROR
Nov 19 23:20:43 [10.241.100.55.4.0] Domain-A.SC [ID 687150 local1.error] RP0
encountered the first error
Nov 19 23:20:43 [10.241.100.55.4.0] Domain-A.SC [ID 295740 local1.error]
/partition0/RP0/dx1:
>>> General Error Status[0x1e] : 0x00008001
             FirstError [15:15] : 0x1
               CPerr [00:00] : 0x1 Control Parity Error
    Safari Port Error Status3[0x22] : 0x00000004
        ScU3IFPar [02:02] : 0x1 Fireplane u3 input FIFO parity error (ports 0
and 2 only)/Input parity error (other ports)
Nov 19 23:20:43 [10.241.100.55.4.0] Domain-A.SC [ID 541909 local1.crit] DomainA
is currently paused due to an error.  This domain must be turned off
via"setkeyswitch off" to recover

Nov 19 23:20:43 [10.241.100.55.4.0] Domain-A.SC [ID 944525 local1.error] CPU
Board at /N0/SB2 DX 2 Data parity error on connection to Repeater Board
(F3800)Slot 10. Domain will continue operation.

Nov 19 23:20:43 [10.241.100.55.4.0] Domain-A.SC [ID 277029 local1.error]
Potentially Damaged Boards

  Sunfire Centerplane
  RP0
  /N0/SB2

Nov 19 23:20:43 [10.241.100.55.4.0] Domain-A.SC [ID 175616 local1.error] CPU
Board at /N0/SB2 DX 3 Data parity error on connection to Repeater Board (F3800)
Slot 10. Domain will continue operation.

Nov 19 23:20:43 [10.241.100.55.4.0] Domai
n-A.SC [ID 277029 local1.error]  \nPotentially Damaged Boards \n \n     Sunfire
Centerplane \n  RP0 \n  /N0/SB2

Nov 19 23:20:43 [10.241.100.55.4.0] Domain-A.SC [ID 928118 local1.error] CPCI
I/O board (F3800) at /N0/IB6 DX 1 Data parity error on connection to Repeater
Board (F3800) Slot 10. Domain will continue operation.

Nov 19 23:20:43 [10.241.100.55.4.0] Domain-A.SC [ID 633100 local1.error]
\nPotentially Damaged Boards \n \n      Sunfire Centerplane \n  RP0 \n  /N0/IB6
See CODE EXAMPLE 4-29 for continuation
```

CODE EXAMPLE 4-29 System Error Followed by Various Parity Errors (continued from CODE EXAMPLE 4-28)

```
Nov 19 23:20:43 [10.241.100.55.4.0] Domain-A.SC [ID 928118 local1.error] CPCI
I/O board (F3800) at /N0/IB6 DX 1 Data parity error on connection to Repeater
Board (F3800) Slot 10. Domain will continue operation.

Nov 19 23:20:43 [10.241.100.55.4.0] Domain-A.SC [ID 633100 local1.error]
\nPotentially Damaged Boards \n \n     Sunfire Centerplane \n  RP0 \n  /N0/IB6

Nov 19 23:20:43 [10.241.100.55.4.0] Domain-A.SC [ID 283272 local1.error] CPCI
I/O board (F3800) at /N0/IB8 DX 1 Data parity error on connection to Repeater
Board (F3800)
Slot 10. Domain will continue operation.

Nov 19 23:20:43 [10.241.100.55.4.0] Domain-A.SC [ID 633102 local1.error]
\nPotentially Damaged Boards \n \n     Sunfire Centerplane \n  RP0 \n  /N0/IB8

Nov 19 23:20:43 [10.241.100.55.4.0] Domain-A.SC [ID 283272 local1.error] CPCI
I/O board (F3800) at /N0/IB8 DX 1 Data parity error on connection to Repeater
Board (F3800) Slot 10. Domain will continue operation.

Nov 19 23:20:43 [10.241.100.55.4.0] Domain-A.SC [ID 633102 local1.error]
\nPotentially Damaged Boards \n \n     Sunfire Centerplane \n  RP0 \n  /N0/IB8

Nov 19 23:20:43 [10.241.100.55.4.0] Domain-A.SC [ID 806392 local1.error]
Repeater Board (F3800) Slot 10 DX 1 Data parity error on connection to CPU
Board at /N0/SB2. Domain will continue operation.
Nov 19 23:20:43 [10.241.100.55.4.0] Domain-A.SC [ID 332481 local1.error]
\nPotentially Damaged Boards \n \n    Sunfire Centerplane \n  /N0/SB2 \n    RP0
```

Analysis

First message seeems to implicate RP0, SB3 or centerplane. However, I don't understand the later messsages.... New version of Scapp?

Suspect FRUs

- RP0
- SB3
- Centerplane

4.7 Remaining SBBC ASIC Errors

The SBBC is used on the SC (1x), System board (2x) and I/O boards (1x).

Q: Why then, do some error messages indicate errors from a repeater SBBC ASIC?

This section excludes Console Bus, Safari, and DCDS errors which are covered in previous sections and lock step errors which are not implemented.

There are two remaining categories of errors:

- SBBC Error Status (see Serengeti APR 3.3.2.3)
- SBBC PCI Error Status (see Serengeti APR 3.3.2.4)

4.7.1 SBBC Error Status

These errors are flagged by the identifier `ErrorStatus` in the error message output.

TABLE 4-3 SBBC Error Status Register Bits

Error bit	Description
LOCKSTEP_ERR	(Not implemented)
SYS_ERR_P1SC	SC only
SYS_ERR_P0SC	SC only
SAF_ERR	L1 boards only
BB_TO	L1 boards only
DCDS_ERR	L1 boards only
BB_ILL	Boot bus illegal access, L1 boards only
BB_ERR	Boot bus error, L1 boards only

4.7.1.1 Example of SB2 SYSTEM ERROR/SBBC ErrorStatus/BB_ILL

The following example (boot bus illegal access) is from EMEA Radiance Case #36399725:

CODE EXAMPLE 4-30 SB2 SYSTEM ERROR/SBBC ErrorStatus/BB_ILL

```
Oct 01 10:04:50 sc0a Domain-B.SC: Domain B has a SYSTEM ERROR
Oct 01 10:04:50 sc0a Domain-B.SC: /N0/SB2 encountered the first error
Oct 01 10:04:50 sc0a Domain-B.SC: RepeaterSbbcAsic reported first error on
/N0/SB2
Oct 01 10:04:50 sc0a Domain-B.SC: /partition0/domain1/SB2/bbcGroup1/sbbc1:
>>> ErrorStatus[0x80] : 0x80008004
                      FE [15:15] : 0x1
                   BBIll [03:02] : 0x1 Illegal access on the bootbus
                  ErrSum [31:31] : 0x1

Oct 01 10:04:51 sc0a Domain-B.SC: Domain B is currently paused due to an error.
This domainmust be turned off via "setkeyswitch off" to recover.
```

Analysis

Since the boot bus is contained entirely on an SB board, any errors would be caused by the board reporting the error or it's CPUs. Thus, SB2 is the suspected FRU for these errors.

Suspect FRU

- SB2

4.7.1.2 Example of SYSTEM ERROR/SDC SafariPortError/SBBC BB_ERR on SB Board

The following example (boot bus error) is from APAC Radiance Case 62556779

CODE EXAMPLE 4-31 SYSTEM ERROR/SDC SafariPortError/SBBC BB_ERR on SB Board

```
Jun 28 22:16:52 server Domain-A.SC [ID 194671 local1.crit] Domain A has a SYSTEM
ERROR
Jun 28 22:16:53 server Domain-A.SC [ID 548884 local1.error] /N0/SB2 encountered
the first error
Jun 28 22:16:53 server Domain-A.SC [ID 965218 local1.error] RepeaterSbbcAsic
reported first error on /N0/SB2
Jun 28 22:16:53 server Domain-A.SC [ID 605200 local1.error]
/partition0/domain0/SB2/sdc0:
>>> SafariPortError[0x200] : 0x00018001
                    FE [15:15] : 0x1
          AccParBidiErr [16:16] : 0x1
            ParBidiErr [00:00] : 0x1 ParityBidi error
>>> SafariPortError1[0x210] : 0x00018001
                     FE [15:15] : 0x1
          AccParBidiErr [16:16] : 0x1
             ParBidiErr [00:00] : 0x1 ParityBidi error
Jun 28 22:16:53 server Domain-A.SC [ID 203561 local1.error]
/partition0/domain0/SB2/bbcGroup0/sbbc0:
>>> ErrorStatus[0x80] : 0x80008002
                  BBErr [01:00] : 0x2 Bootbus command or arbitration protocol
error detected for ports
                     FE [15:15] : 0x1
                 ErrSum [31:31] : 0x1
Jun 28 22:16:53 server Domain-A.SC [ID 541909 local1.crit] Domain A is currently
paused due to an error.  This domain must be turned off via "setkeyswitch off"
to recover
```

Analysis

Since the boot bus is contained entirely on an SB board, any errors would be the result of the board reporting the error or one of it's CPUs. Thus, SB2 is the suspect FRU for these errors.

Suspect FRU

- SB2

4.7.1.3 Example of SB SYSTEM ERROR/SBBC ErrorStatus/BBIll

The following example (illegal boot bus access) is from EMEA Radiance Case #36399725:

CODE EXAMPLE 4-32 SB SYSTEM ERROR/SBBC ErrorStatus/BBILL

```
Oct 01 10:04:50 sc0a Domain-B.SC: Domain B has a SYSTEM ERROR
Oct 01 10:04:50 sc0a Domain-B.SC: /N0/SB2 encountered the first error
Oct 01 10:04:50 sc0a Domain-B.SC: RepeaterSbbcAsic reported first error on
/N0/SB2
Oct 01 10:04:50 sc0a Domain-B.SC: /partition0/domain1/SB2/bbcGroup1/sbbc1:
>>> ErrorStatus[0x80] : 0x80008004
                      FE [15:15] : 0x1
                   BBIll [03:02] : 0x1 Illegal access on the bootbus
                  ErrSum [31:31] : 0x1
Oct 01 10:04:51 sc0a Domain-B.SC: Domain B is currently paused due to an error.
This domain must be turned off via "setkeyswitch off" to recover
```

Analysis

Since the boot bus is contained entirely on an SB board, any errors would be the result of the board reporting the error or one of it's CPUs. Thus, SB2 is the suspect FRU for these errors.

Suspect FRU

- SB2

4.7.2 SBBC PCI Error Status

You can identify these errors by the string PCI ErrorStatus in the output.

Note: These errors can only occur on the SC and IB boards (not used on SB boards).

NOTE: The SBBC PCI interface is connected to only Leaf B of Schizo #0 of an I/O board (all three types?). This may be important in troubleshooting failures.

TABLE 4-4 PCI Error Status Register Bits

Error bit	Description
S_ILL	Slave port detected illegal access
S_PERR	Slave port detected a parity error
M_SERR	Master port detected a system error
M_PERR	Master port detected a parity error

No guidance from troubleshooting guide....

What is Console not idle or BB not idle errors in Tshoot guide? Not in SAPR!

4.7.2.1 Example of SYSTEM ERROR/SBBC-ErrorStatus/PCIErrorStatus/SIll

The source for this example was AMER/Radiance Case #62670285

CODE EXAMPLE 4-33 SYSTEM ERROR Followed by SBBC PCIErrorStatus

```
Sep 21 10:49:43 sc0 Domain-C.SC: Domain C has a SYSTEM ERROR
Sep 21 10:49:43 sc0 Domain-C.SC: /N0/IB8 encountered the first error
Sep 21 10:49:43 sc0 Domain-C.SC: RepeaterSbbcAsic reported first error on
/N0/IB8
Sep 21 10:49:43 sc0 Domain-C.SC: /partition1/domain0/IB8/bbcGroup0/sbbc0:
    ErrorStatus[0x80] : 0x80000000
                  ErrSum [31:31] : 0x1
>>> PCIErrorStatus[0xe0] : 0x00008020
                      FE [15:15] : 0x1
                    SIll [05:05] : 0x1 Slave port detected an illegal access

Sep 21 10:49:43 sc0 Domain-C.SC: Domain C is currently paused due to an error.
This domain must be turned off via "setkeyswitch off" to recover.
```

Analysis

Question: What is an illegal access?

Can we attribute any of these to the SBBC itself, or any cards on LEAF B of Schizo #0?

What about the address captured? Can we trace it to a card?

In any case, the failed FRU is the board reporting the error (per Table 3-12) in the Serengeti Architecture PR, in this case IB8. However, several Radiance cases note different solutions. In AMER Radiance CASE #62562513, the solution was the replacement of 'FC cPCI' cards (501-5570) in both systems.

In AMER Radiance CASE #???: misconfigured FC card:

*** jnic.conf variables set

FcLoopEnabled =0;

FcFabricEnabled =1;

FcPortCfgEnable=1;

Suspect FRUs:

4.7.3 Console Bus Hub (CBH) ASIC Errors

CBH ASIC is only on the SC.... Refer also to console bus errors..

Move to SC section?.

4.7.4 Remaining DCDS ASIC Errors

The DCDS (Dual Cheetah Data Switch) ASIC is only used on the SB boards.

This section excludes CPU and Safari-Side data parity errors (covered in Chapter 2) and lockstep errors which are not implemented. The remaining category of errors are Cheetah0/1 (see Serengeti APR 8.3.7).

DCDS errors are reported by the SBBC ASIC and can be located by the following message identifiers:

- `RepeaterSbbcAsic`
- `DcdsErr`
- `Cheetah0ErrorStatus` or `Cheetah1ErrorStatus`

The following error bits are for the Cheetah Error Status Register.

- _LOCKSTEP_ERRMDATA
- CDATA_LOCKSTEP_ERR
- SSEL_LOCKSTEP_ERR
- MSEL_LOCKSTEP_ERR
- CSEL_LOCKSTEP_ERR
- C0_DPERRCovered in section #1
- SAF_DPERRCovered in section #1
- S2C_BYPASS
- M2C_BYPASS
- M2S_FIFO_OFLO_ERR
- S2C_FIFO_OFLO_ERR
- M2C_FIFO_OFLO_ERR
- M2S_FIFO_UFLO
- S2C_FIFO_UFLO
- M2C_FIFO_UFLO

4.7.5 Example of DCDS M2CBypass Error From an SB

The following example is from Radiance Case 62344116.

CODE EXAMPLE 4-34 DCDS M2CBypass error From an SB

```
eb 06 03:26:14 serengeti-sc0 Domain-C.SC: Domain C has a SYSTEM ERROR
Feb 06 03:26:15 serengeti-sc0 Domain-C.SC: /N0/SB4 encountered the first error
Feb 06 03:26:15 serengeti-sc0 Domain-C.SC: RepeaterSbbcAsic reported first
error on /N0/SB4
Feb 06 03:26:15 serengeti-sc0 Domain-C.SC:
/partition1/domain0/SB4/bbcGroup1/sbbc1:
>>> ErrorStatus[0x80] : 0x80008010
                 DcdsErr [04:04] : 0x1 DCDS asserted an error
                      FE [15:15] : 0x1
                  ErrSum [31:31] : 0x1

Feb 06 03:26:15 serengeti-sc0 Domain-C.SC:
/partition1/domain0/SB4/bbcGroup1/cpuCD/dcds4:
>>> Cheetah1ErrorStatus[0x61] : 0x00008040
               M2CBypass [06:06] : 0x1 Cheetah 1 attempted to bypass Memory 0
while M12C1 fifo was not empty
                      FE [15:15] : 0x1

Feb 06 03:26:15 serengeti-sc0 Domain-C.SC: Domain C is currently paused due to
an error.  This domain must be turned off via "setkeyswitch off" to recover
```

4.7.5.1 Analysis

All of the DCDS errors can be attributed to one of the following:

- A CPU
- CPU socket
- SB4
- SB4 DCDS ASICs.

In any case, these errors all point to SB4 as the faulty FRU.

4.7.5.2 Suspect FRU

- SB4

4.8 System Clock Errors

Need examples

CHAPTER 5

Environmental Errors

5.1 Environmental Errors

5.1.1 Example of IB Board Abnormal Temperatures

The source for the following example is from APAC Radiance Case 10142028.

CODE EXAMPLE 5-1

```
Nov 10 03:37:11 eqmfire2 Chassis-Port.SC [ID 110676 local0.error] Device
voltage problem: /N0/IB6 abnormal state for device: Board 0 1.5 VDC 0 Value:
0.05 Volts DC
Nov 10 03:37:11 eqmfire2 Chassis-Port.SC [ID 272150 local0.error] Device
voltage problem: /N0/IB6 abnormal state for device: Board 0 3.3 VDC 0 Value:
0.56 Volts DC
Nov 10 03:37:12 eqmfire2 Chassis-Port.SC [ID 298237 local0.error] Device
voltage problem: /N0/IB6 abnormal state for device: Board 0 5 VDC 0 Value: 0.42
Volts DC
Nov 10 03:37:12 eqmfire2 Chassis-Port.SC [ID 277529 local0.error] Device
voltage problem:
/N0/IB6 abnormal state for device: Board 0 12 VDC 0 Value: 0.55 Volts DC
```

5.1.2 Analysis

5.1.3 Suspect FRU

- IB6

5.1.4 Example of SB Board Abnormal Temperatures

The following example is of a `showlogs` command output that indicates that board 5 on a system is shut down due to a temperature problem on one of the CPUs.

CODE EXAMPLE 5-2 showlogs Command Temperature Error Message

```
serengeti-sc0:SC> showlogs
          Dec 12 08:31:00 serengeti-sc0 Chassis-Port.SC: Domain A has a SYSTEM
ERROR
           Dec 12 08:31:07 serengeti-sc0 Chassis-Port.SC: This domain is still
running because error pause is not enabled for this domain
          Dec 12 08:31:12 serengeti-sc0 Chassis-Port.SC: FULL-COOLING ENGAGED.
Turning fans high
           Dec 12 08:31:13 serengeti-sc0 Chassis-Port.SC: FT0, fan speed, High
(4,2)
           Dec 12 08:31:14 serengeti-sc0 Chassis-Port.SC: FT1, fan speed, High
(4,2)
           Dec 12 08:31:15 serengeti-sc0 Chassis-Port.SC: FT2, fan speed, High
(4,2)
           Dec 12 08:31:17 serengeti-sc0 Chassis-Port.SC: FT3, fan speed, High
(4,2)
            Dec 12 08:31:18 serengeti-sc0 Chassis-Port.SC: Device temperature
problem: /N0/SB5 auto power off may occur due to device: Cheetah 3 Temp. 0 Value:
127 Degrees C
            Dec 12 08:31:19 serengeti-sc0 Chassis-Port.SC: Device temperature
problem: Shutting down /N0/SB5 due to temperature of device: Cheetah 3 Temp. 0
Value: 127 Degrees C
            Dec 12 08:31:19 serengeti-sc0 Chassis-Port.SC: /N0/SB5, sensor
status, over limit (7,1,0x201050603030000)
          Dec 12 08:32:08 serengeti-sc0 Chassis-Port.SC: ...board successfully
powered off.
            Dec 12 08:32:08 serengeti-sc0 Chassis-Port.SC: /N0/SB5, sensor
status, shutdown (7,3,0x201050603030000)
            Dec 12 08:34:42 serengeti-sc0 Chassis-Port.SC:  Turning fans low

           Dec 12 08:34:42 serengeti-sc0 Chassis-Port.SC: FT0, fan speed, Low
(4,1)
           Dec 12 08:34:44 serengeti-sc0 Chassis-Port.SC: FT1, fan speed, Low
(4,1)
           Dec 12 08:34:45 serengeti-sc0 Chassis-Port.SC: FT2, fan speed, Low
(4,1)
           Dec 12 08:34:47 serengeti-sc0 Chassis-Port.SC: FT3, fan speed, Low
(4,1)
```

5.1.5 Suspect FRU

- SB5