

93/02/22
07:51:00

(trngps)(mikey:throg Job: vol.8 Date: Mon Feb 22 07:50:58 1993)

psfilter.in.23447

1

6 JAN 93
Revision 8

C3800 Troubleshooting Guide

C3800
TROUBLESHOOTING
HINTS

Chapter 1 OPUS

1. OPUS Memory Component Identification
2. OPUS Software Revision Information
3. Setting up the OPUS Modem
4. OPUS Screen Characteristics and Network Use
5. Removing OPUS Printer Cable
6. Multiple Screens on the Opus Workstation
7. Changing the Timezone on Opus Workstation
8. Opus Depends on C3800 Free Running Clocks
9. C3800 Modem Replacement Procedure
10. C3800 Spu Console Window Messages
11. OPUS Status Utility "sfp"
12. OPUS Backup and Restore MODIFIED
13. Defective Keyswitches
14. Loading 2.0 SPUOS
15. Auto Reboot Script Warning
16. SPU Prompts
17. SPU Software Restore
18. SPU Utility Execution
19. XSFP Keyswitch positions and power-up actions
20. Spuio Problem NEW
21. SPU tape boot failure NEW

Chapter 2 Diagnostics

1. C3800 PETS PROBLEMS/HINTS
 - A. Loading PETS thru SPU
 - B. PETS Startup Problem
 - C. PETS Version 3.0
2. C3800 Continuity Test After Board Replacement
3. C3800 pram.log Created by New hard_logger Dumpers
4. C3800 Diagnostic Bugs
5. Verifying C3800 Control Stores
6. Multi-head Troubleshooting Utilities
7. C3800 spu4000 connectivity test (signals not covered in current release)
8. C3800 Net Resistance Value
9. Continuity Test Failure Before sysreset
10. IA Soft Errors and rslog Utility
11. Remote execution of diags spu4000, cu4000 and mem4000
12. Error Rules for CXTS
13. SST Discussion
 - A. Version 2.2.3
 - B. Version 3.0
14. DDB Primer
15. Scan based test failures with rev H NSP's
16. Diagnostics Version 3.0.0.6
17. Spu4000 Faults
18. Spu4000 Bug
19. Rev H NSP Failures with 2.0 Diagnostics
20. Diagnostics Version 3.1
21. SST Version 3.1 REVIEW
22. Spu4000 and intermittent failures NEW

93/02/22
07:51:00

(trngps)(mikey:throg Job: vol.8 Date: Mon Feb 22 07:50:58 1993)

psfilter.in.23447

2

C3800 Troubleshooting Guide

Chapter 3 C3800

1. C3 Bay Configuration Considerations
2. C3 I/O Bulkhead Boards
3. C3 Purge Ram Function
4. C3800 Gate Array Functions
5. New hard dumpers for known failures
6. C3800 Firmware Update Procedures:
7. Displaying The C3800 System Serial Number
8. C3800 osc_freq and Released NCU's
9. C3800 STRAMS
10. C3800 pram.log Created by New hard_logger Dumpers
11. C3800 Front Panel Switch Problem
12. C3800 diaginit abort
13. C3800 Gate Array Identification
14. NCU Powered off and mminit Failures
15. Memory Initialization Failures
16. NMB Failure Precautions
17. Crossbar/NCU Power Removal Warning
18. NSP Revision History
19. C3800 Hard Error List
20. xsysconfig reminder
21. Powered down Bay Detection
22. Removing Defective Boards
23. C3800 wredc parity errors at upper clock
24. Undetected NMB deallocation
25. New Boards
26. Restarting C3800 after crash
27. Distinguishing STRAM and Purge Ram Failures
28. Memory Configuration Related Crashes
29. Memory Errors with Hitachi Dram's
30. Identifying STRAM and PRAM Failures
31. NVRF2 Failures
32. New Keyswitch
33. NCU related hard error: init_commregs
34. Potential Epont Problems
35. Update Data Base
36. ECC Timed Backoff Errors
37. Crashdump validity

REVIEW
REPLACED

NEW
NEW
NEW

Chapter 4 SWIS/SWIP

1. swis Failure Indications
2. Isolating swip and NCU failures
3. swip/swis Register Bit Definitions
4. Swis Error Processing
5. Swip Error Processing
6. Intermittent swis errors

NEW

C3800 Troubleshooting Guide

Chapter 5 Miscellaneous

1. C3800 Transformer Requirement
2. C3800 Board Damage During Shipping
3. C3800 Crossbar Power Pallet Nutplates
4. Multiple Boot Scripts
5. Kernel Damage during Virgin Software Install
6. Criteria for board replacement
7. Crossbar Backplane Replacement Procedure
8. 10.2 OS patches

Chapter 6 Power

1. Power Brick Locations
2. C3800 PPC Brick Fuses
3. C3800 Crossbar Power Pallets
4. C3800 Power Problem Isolation
5. C3800 Voltage Margins
6. EPROM damage to CCU's when Powering Down NIA
7. C3800 Power Transformer Specification
8. Site Prep Guide Power
9. PPC and BPC failures
10. Board Temperature Sensors
11. Using pwr_util
12. Isolating Crossbar Power Problems
13. CCU Temperature sensors

NEW

Chapter 7 I/O

1. CCU Revision Levels
2. IA soft errors
3. I/O Bulkhead Connectors
4. Loading TLI4480
5. HiPPI Installation Prereq
6. C3800 Pbus Architecture
7. io5000 failures
8. Failures related to FDDI
9. hp14000 Diagnostic Failures
10. IDC Related ECC Errors
11. VIOP Memory Pages for Ethernet
12. Installing ITC
13. VIOP Memory Pages
14. Idcfmt failure on SPU

NEW

NEW

NEW

C3800 Troubleshooting Guide

Chapter 8 Utilities and Registers

- 1. Utilities List
- 2. Register List
- 3. APR Utility
- 4. Debug Scripts
- 5. Continuity Tests and Functions
- 6. Memory soft Tools
- 7. Display_log Utility
- 8. Utility Definitions
- 9. Cop Utility
- 10. Unsupported Utilities
- 11. APR, failure to clear Harderr
- 12. CXTS
- 13. Crashdump -H problem REVIEW
- 14. Head Offline on Reboot NEW
- 15. APR start script NEW

Chapter 9 Troubleshooting

- 1. NVRF Hard Error Explanation
- 2. NDAT Error Hints REVIEW
- 3. Diagnosing NMB Failures
- 4. Isolating System Hangs
- 5. NVRF2 Crash Troubleshooting
- 6. Investigating diaginit failures
- 7. Locating Connectivity Failures
- 8. ucode_pulled_hard_err analysis
- 9. wredc_par_err analysis
- 10. SPU Tape boot failure NEW

C3800 Troubleshooting Guide

Table of Contents

CHAPTER I 1
(OPUS)

- 1.* OPUS Memory Component Identification 1
- 2.* OPUS Software Revision Infomation 3
- 3.* Setting up the OPUS Modem 4
- 4.* OPUS Screen Characteristics and Network Use 4
- 5.* Removing OPUS Printer Cable 7
- 6.* Multiple Screens on the Opus Workstation 7
- 7.* Changing the Timezone on Opus Workstation 7
- 8.* Opus Depends on C3800 Free Running Clocks 8
- 9.* C3800 Modem Replacement Procedure 8
- 10.* C3800 Spu Console Window Messages 9
- 11.* OPUS Status Utility "sfp" 9
- 12.* OPUS Backup and Restore 10
- 13.* Defective C3800 Keyswitches 14
- 14.* Installing C3800 SpuOS 2.0 14
- 15.* Auto Reboot Script Warning 14
- 16.* Spu Prompts 15
- 17.* SPU Software Restore 15
- 18.* SPU Utility Execution 15
- 19.* XSFP Key-Switch Positions and Power-up Actions 15
- 20.* Spuio Problem 17
- 21.* Spu Tape boot failure 17

CHAPTER II 19
(DIAGNOSTICS)

- 1.* C3800 PETS PROBLEMS/HINTS 19
- 2.* C3800 Continuity Test After Board Replacement 21
- 3.* C3800 pram.log Created by New hard_logger Dumpers 21
- 4.* C3800 Diagnostic Bugs 22
- 5.* Verifying C3800 Control Stores 22
- 6.* Multi-head Troubleshooting Utilities 22
- 7.* C3800 spu4000 connectivity test (signals not covered
in 23
- 8.* C3800 Net Resistance Value 24
- 9.* Continuity Test Failure Before sysreset 24
- 10.* IA Soft Errors and rslog Utility 25
- 11.* Remote Execution of diags spu4000, cu4000 and
mem4000 25
- 12.* Error Rules for CXTS 31
- 13.* SST Discussion 38
 - A.* Version 2.2.3 41
 - B.* Version 3.0 41
- 14.* DDB Primer 43
- 15.* Scan Based Test Failures With REV H NSP's 54
- 16.* C3800 Diagnostics Version 3.0.0.6 55
- 17.* spu4000 Faults 58
- 18.* spu4000 Bug 58
- 19.* Rev H NSP failures With 2.0 Diagnostics 58
- 20.* Diagnostics version 3.1 59
- 21.* SST Version 3.1 62
- 22.* spu4000 and Intermittent Failures 62

CHAPTER III 63
(C3800)

- 1.* C3 Bay Configuration Considerations 63
- 2.* C3 I/O Bulkhead Boards 63
- 3.* C3 Purge Ram Function 64

C3800 Troubleshooting Guide

- 4.* C3800 Gate Array Functions 64
- 5.* New hard dumpers for known failures 65
- 6.* C3800 Firmware Update Procedures: 66
- 7.* Displaying The C3800 System Serial Number 66
- 8.* C3800 osc_freq and Released NCU's 66
- 9.* C3800 STRAMS 67
- 10.* C3800 pram.log Created by New hard_logger Dumpers 68
- 11.* C3800 Front Panel Switch Problem 69
- 12.* C3800 diaginit abort 69
- 13.* C3800 Gate Array Identification 69
- 14.* NCU Powered off and mminit Failures 70
- 15.* Memory Initialization Failures 70
- 16.* NMB Failure Precautions 70
- 17.* Crossbar/NCU Power Removal Warning 71
- 18.* NSP Revision History 71
- 19.* C3800 Hard Error List 72
- 20.* xsysconfig reminder 74
- 21.* Powered Down Bay Detection 74
- 22.* Removing Defective Boards 75
- 23.* C3800 wredc Parity Errors at Upper Clock 75
- 24.* Undetected NMB deallocation 75
- 25.* C3800 New Boards 76
- 26.* Restarting a C3800 After a Crash 77
- 27.* Distinguishing STRAM and Purge Ram Failures 77
- 28.* Memory Cofiguration related Crashes 78
- 29.* Memory Errors 79
- 30.* Identifying STRAM and PRAM Failures 79
- 31.* NVRF2 Failures 82
- 32.* New Keyswitch 84
- 33.* NCU related error: init_commregs 85
- 34.* Potential epont Problem on C3800 86
- 35.* Updating 3800 Data Base 86
- 36.* ECC Timed Backoff Error Message 87
- 37.* Crashdump Validity 88
- CHAPTER IV 89
- (SWIP/SWIS)
- 1.* swis Failure Indications 89
- 2.* Isolating swip and NCU failures 89
- 3.* swip/swis Register Bit Definitions 90
- 4.* SWIS Error Processing 92
- 5.* SWIP Error Processing 93
- 6.* Intermittent swis Errors 95
- CHAPTER V 96
- (MISC)
- 1.* C3800 Transformer Requirement 96
- 2.* C3800 Board Damage During Shipping 96
- 3.* C3800 Crossbar Power Pallet Nutplates 96
- 4.* Multiple Boot Scripts 97
- 5.* Kernel Damage During a Virgin Install 97
- 6.* Criteria for Board Replacement 98
- 7.* Replacing the Crossbar Backplane 98
- 8.* ConvexOs 10.2 patches 99
- CHAPTER VI 100
- (POWER)
- 1.* Power Brick Locations 100
- 2.* C3800 PPC Brick Fuses 105
- 3.* C3800 Crossbar Power Pallets 106

C3800 Troubleshooting Guide

- 4.* C3800 Power Problem Isolation 106
- 5.* C3800 Voltage Margins 110
- 6.* EPROM Damage When Powering Down A NIA Board 111
- 7.* C3800 Power Transformer Specification 111
- 8.* Site Prep Guide Power 113
- 9.* PPC and BPC Failures - EEPROM Not Write Protected 113
- 10.* Board Temperature Sensors 114
- 11.* Using pwr_util 115
- 12.* Isolating Crossbar Power Problems 117
- 13.* CCU Temperature Sensors 117
- CHAPTER VII 118
- (I/O)
- 1.* CCU Revision Levels 118
- 2.* C3800 IA Soft Errors 118
- 3.* I/O Bulkhead Connectors 119
- 4.* Loading tli4480 120
- 5.* HiPPI Installation Prereq 120
- 6.* C3800 Pbus Architecture 120
- 7.* io5000 Failure on C3800 120
- 8.* Failures related to FDDI 121
- 9.* hpi4000 Diagnostic Failure 121
- 10.* IDC related ECC errors 122
- 11.* VIOP Memory Pages With Ethernet 122
- 12.* Installing an ITC 122
- 14.* Idcfmt failure on SPU 123
- CHAPTER VIII 124
- (Utilities and Registers)
- 1.* C3800 Utilities List 124
- 2.* C3800 Registers 124
- 3.* C3800 APR Utility 125
- 4.* Debug Scripts 129
- 5.* C3800 Continuity Tests and Functions 129
- 6.* Memory Soft Error tools 130
- 7.* display_log Utility 132
- 8.* Utility Definitions 133
- 9.* cop Utility 133
- 10.* Unsupported Utilities 134
- 11.* APR failure to clear Harderr 134
- 12.* CXTS - Convex Xpert Troubleshooting System 135
- 13.* Crashdump -H problem 136
- 14.* Head offline on reboot 136
- 15.* ARP Start Script 137
- CHAPTER IX 138
- (Troubleshooting)
- 1.* NVRF Hard Error Explanation 138
- 2.* NDAT Error Hints 141
- 3.* Diagnosing NMB Failures 142
- 5.* nvrf2 Crash Troubleshooting 145
- 6.* Investigating diaginit Failures 146
- 7.* Locating Connectivity Problems 147
- 8.* ucode_pulled_hard_err analysis 147
- 9.* wredc_par_err analysis 148
- 10.* Using rcque to find data 153

C3800 Troubleshooting Guide

CHAPTER I
(OPUS)

1.* OPUS Memory Component Identification

When encountering a memory error during self test the OPUS workstation for the C3800 does not directly display the failing module. This bulletin should help in isolating the failed component.

The normal error on the OPUS will be of the type BxBy where the relationship to possible simms is shown in the table below:

x,y	SIMMS
0 0	MR8, MR4, ML8
0 1	MR7, MR3, ML7
0 2	MR6, MR2, ML6
0 3	MR5, MR1, ML5
1 0	MR8
1 1	MR7
1 2	MR6
1 3	MR5
2 0	ML4
2 1	ML3
2 2	ML2
2 3	ML1
3 0	MR4
3 1	MR3
3 2	MR2
3 3	MR1

In cases where further isolation is necessary it will be necessary to use the physical address. The physical address can be acquired from the syslog file located in /var/adm/messages. The format of the entry for the memory error is as follows:

date time info: pme=nnnnnnnn,phys addr=wnnnnnz

This physical address is only necessary if the x value is 0 in the error display BxBy. If this is the case then use the following table:

phys addr	x,y	SIMM
nnnnnn	0 0	ML8
where leading 0	0 1	ML7
is dropped in 7	0 2	ML6
digit addr.	0 3	ML5
1nnnnnn	0 0	MR8
where addr has	0 1	MR7
leading 1	0 2	MR6
	0 3	MR5
3nnnnnn	0 0	MR4
where addr has	0 1	MR3
leading 3	0 2	MR2
	0 3	MR1

C3800 Troubleshooting Guide

If, for some reason, the error message is not displayed, the faulty SIMM can be isolated from the physical address as follows:

phys addr	where z =			
	0,4,8,c	1,5,9,d	2,6,a,e	3,7,b,f
nnnnnz leading 0	ML8	ML7	ML6	ML5
1nnnnnz	MR8	MR7	MR6	MR5
2nnnnnz	ML4	ML3	ML2	ML1
3nnnnnz	MR4	MR3	MR2	MR1

Here are some examples:

EXAMPLE 1:
Phys Addr: 1394CA8
 ^ ^ ^
 || |__ 1000
 || |__ 00
 || _____ 01nnnnnn
 SIMM: MR8

EXAMPLE 2:
Phys Addr: 3381AB8
 ^ ^ ^
 || |__ 1000
 || |__ 00
 || _____ 03nnnnnn
 SIMM: MR4

EXAMPLE 3:
Phys Addr: 32A0204
 ^ ^ ^
 || |__ 0100
 || |__ 00
 || _____ 03nnnnnn
 SIMM: MR4

2.* OPUS Software Revision Information

To locate the revision levels of current software on the OPUS, list the following files:

- /SPUOS_REV
- /diag/DIAG_REV
- /mnt/os/VMUNIX_REV

93/02/22
07:51:00

(trngps)(mikey:throg Job: vol.8 Date: Mon Feb 22 07:50:58 1993)

psfilter.in.23447

6

C3800 Troubleshooting Guide

3.* Setting up the OPUS Modem

This is the procedure for getting the modem functional on the OPUS. This should not be necessary as all should be done in the software, but in case a more indepth explanation is necessary.

Each time the rc.local script is run, the modem is initialized with the command '/diag/bin/modem_init'. All this does is open the modem tty, and send several AT commands to it. The commands that it sends are simply listed in the text file "/diag/db/modem_parms". If the modem is not the FasTalk, some of these commands may need to change. There are three settings that are important.

The modem needs to be in auto-answer.

On the FasTalk this was accomplished with a "ats0=1"

The modem needs to talk to the tty at 9600 baud, regardless of the connect speed (this is usually the default). This had to be set on the FasTalk because its default was a thing called 9600 trellis mode, which does not work well with any modem other than a FasTalk. On a FasTalk the command that does this is "at%b9600".

The modem needs to use the carrier detect line. Most modems have this defaulted to turn the line on and leave it on - if that is the case, getty will grab the tty line and try to talk to it (forever). This causes logins to constantly time out, and you won't be able to dial out. The command that does this on the FasTalk was "at&c1".

It is also noted that the operating mode is set to Normal, this turns off error correction. This is not desirable, but at any rate "at\n0" is the command to enable this.

4.* OPUS Screen Characteristics and Network Use

As follow are the characteristics of the Opus screens, on the C3800, and the functionality of using them on a network:

The xsfp is the program with the representation of the keyswitch. It does a lot of little things; it provides buttons for booting, a toggle for the printer, and a radio box for the boot mode. But its main job is to provide SPU security. It restricts workstation keyboard and mouse IO when the keyswitch is in the SECURE position. For this reason, if it goes away, so does X. It is this feature that makes the xsfp non-restartable.

The xsfp receives interrupts from the hardware whenever the state of the keyswitch changes. It uses this information to determine when to boot OS, and when to restrict user input. These interrupts are only sent to one process at a time. For this reason, there can only be one xsfp running at a time.

C3800 Troubleshooting Guide

The CONVEXOS CONSOLE is the window that CONVEXOS should always be booted from. It is basically an xterm, but has added features that make it unique. It provides the functionality to dump characters written to the screen to a printer. The CONVEXOS CONSOLE also allows a user, logged in as rmtdiag, to take control of the window from a remote tty. This is explained in greater detail below.

If destroyed, CONVEXOS CONSOLE is restarted automatically by the xsfp. This is fine as long as CONVEXOS is not booted. The problem when CONVEXOS CONSOLE is booted is that the console driver is no longer associated with the window, and since the console driver is not restartable, the system enters a state of limbo. I would expect that any program running on CONVEXOS that would cause the console driver to write, would hang. Much the same affect as doing a control S on a C2 spu terminal. The moral of the story is never kill a window that has CONVEXOS booted in it.

The other window of importance is the SPU CONSOLE. There is absolutely nothing special about this window. It is an xterm that is given the -C flag. This simply means that messages written to /dev/console from programs running on the SPU workstation will appear in this window. Note that this window only displays the /dev/console messages until another xterm is started with a -C option. This can potentially lead to confusion.

But this is supposed to be about remote control of the CONVEX. There are two levels of remote interaction, one is simple, providing access to the CONVEXOS CONSOLE window, the other has a bit more complexity but offers complete control of the SPU from a networked workstation running X.

rmtdiag

To gain control of the CONVEXOS CONSOLE window, one simply logs in as the user rmtdiag. When this user logs in, the CONVEXOS CONSOLE is told which device rmtdiag has logged in through. The log in can be either through a tty port, or from an 'login'. After resizing the CONVEXOS CONSOLE window to match that of rmtdiag's terminal (resizing only works if rmtdiag logs in from a vt100 type terminal or an xterm), CONVEXOS CONSOLE begins communication with that device, and ignores keyboard input from the SPU workstation keyboard. Both the remote screen and the CONVEXOS CONSOLE receive the same output. Since this is the window that CONVEXOS console driver runs in, the familiar ^P, ^D functionality works to give access to CONVEXOS, or the SPU.

When the rmtdiag user is done, he cannot simply log out, since he is actually borrowing control of the window that CONVEXOS is booted in. The only "good" way to give up control is to run "remote disconnect". This notifies CONVEXOS CONSOLE that rmtdiag is done, CONVEXOS CONSOLE quits communicating with the remote port, and again takes input from the SPU workstation keyboard.

93/02/22
07:51:00

(trngps)(mikey:throg Job: vol.8 Date: Mon Feb 22 07:50:58 1993)

psfilter.in.23447

7

C3800 Troubleshooting Guide

Networked SPU

To gain broader control of the SPU, a networked SPU is used. In this case, the same programs are run on the SPU workstation, but their output is displayed on a remote workstation attached by ethernet. This is all accomplished by setting the DISPLAY environment variable in the startup script.

To make the changes, you must have root access.

In diaguser's home directory, the .xinitrc file should be modified in the following manner:

```
mwm &
xsetroot -solid navy
sleep 5 # gotta wait so that mwm gets straight before starting clients
.
.
.
becomes
.
.
.
mwm &
xsetroot -solid navy
DISPLAY=other_workstation_name:0.0
mwm &
xsetroot -solid navy
sleep 5 # gotta wait so that mwm gets straight before starting clients
.
.
.
```

Everything started after the resetting of DISPLAY, appears on the networked workstation. Note that the starting of the second mwm is optional, depending on whether the root display menus provided by the motif window manager are desired on the networked workstation. If the motif window manager is to be started from the SPU workstation, the networked workstation should be running an X server, but not a window manager. Conversely, if the second mwm command is omitted from the above file, the networked workstation should be running both an X server and a window manager.

If it is desired that the SPU CONSOLE window be displayed on the networked workstation, a similar change is needed to /etc/spu_console_start. Remember that this window displays all messages written to /dev/console by programs running on the SPU workstation.

```
/bin/su diaguser -c '/usr/bin/X11/xterm -C -ls -name "SPU CONSOLE" -sl
500 -sb -geometry 80x23+0+0 -display unix:0.0'
```

becomes

```
/bin/su diaguser -c '/usr/bin/X11/xterm -C -ls -name "SPU CONSOLE" -sl
500 -sb -geometry 80x23+0+0 -display other_workstation_name:0.0'
```

C3800 Troubleshooting Guide

Extreme care should be exercised when making this change, only the line shown should be changed.

Once this change is made, simply kill the current SPU CONSOLE window, 'init' will automatically start a replacement with the new display environment variable. Be careful when starting new SPU CONSOLES by hand. Only the "last" xterm started with the -C option will receive console output, confusion will reign if a SPU CONSOLE is started on one display when a SPU CONSOLE window is running on another (the old one simply acts like an xterm).

Additionally, the DISPLAY environment variable is set to unix:0.0 in diaguser's .profile and .cshrc files. These should also be set to identify the workstation that is to provide X services.

5.* Removing OPUS Printer Cable

It is important to understand that the console printer on the Opus is connected via the swis. Removal and replacement of the printer cable should be avoided with power applied, as this can cause damage to the swis board.

6.* Multiple Screens on the Opus Workstation

It is not possible to start a second xsfp screen on the opus as this screen is hard coded to respond to certain interrupts. When a second xsfp is initiated this will force a conflict between the two screens and cause the workstation to crash. Although this will not directly crash the C3800, when the workstation is rebooted, the swis will force a reset of the power subsystem and this will bring the system down.

It is possible to startup a second ConvexOS screen, but it is inadvisable, as the two screens will compete for interrupts. This could lead to chaos as processes would be handled across both screens, making it very difficult to keep a coherent process going. The second screen, should it be necessary, can be started by entering "CONVEXOS_CONSOLE" in any window. Again, it is not recommended, except in an emergency.

7.* Changing the Timezone on Opus Workstation

The easiest way to modify the timezone on the Opus workstation is by executing the utility /usr/etc/install/sys_config. This utility will prompt for pertinent SPU information such as network availability, node name, date/time and time zone the system will exist in. This utility is response driven, so requires no arguments to the command.

It will be necessary to execute this from superuser.

93/02/22
07:51:00

(trngps)(mikey:throg Job: vol.8 Date: Mon Feb 22 07:50:58 1993)

psfilter.in.23447

8

C3800 Troubleshooting Guide

8.* Opus Depends on C3800 Free Running Clocks

It has been discovered that the Opus, or more precisely swip, is dependent on free running clocks on the C3800. If the clocks are interrupted, or the NCU is interrupted for any reason, this can result in a crash of the Opus. Generally this crash will give very few other indications, except for the crash. Some details may be obtained by use of the display_log, or the '/var/adm/messages' file resident on the Opus.

A loose or misfitting swip to ncu cable may also result in clocks being interrupted to the swip and thus create the same type of cr%%[PrinterError: out of paper]%% s%%{!@igintprError: no paper tray }%% ash.

This problem can become particularly problematic during the installation phase, as the clocks and clock cables are in a somewhat unknown state. This possibility should be considered when facing successive Opus related crashes.

NOTE It should be understood that Opus crashes do not always result in a crash of the C3800, until the SPU is rebooted. Rebooting the Opus will result in a system wide reset and bring the system down, hard. Because of this, it is advisable to verify that the system is not up, before rebooting the workstation. If the system is still up then it may be possible to schedule the outage with the customer.

It should, also, be remembered that the swip is subject to thermal failure during the Opus boot process. This type of problem can result in a loss of 10 to twenty minutes while waiting for the transceivers to cool down. More information on this failure is available in Tech Bulletin v3w9.

It should be noted that it is necessary to do a general cleanup of the swip registers after a crash, or a failed attempt to diaginit, or boot. This cleanup can often be accomplished by issuing two successive "cleanup" commands at the spu prompt. If this does not clear all registers it will be necessary to manually change the contents. More information on this is located in Technical Bulletin v3w5.

9.* C3800 Modem Replacement Procedure

To replace the C3800 modem without bring the SPU down, first power down the old modem, replace the old modem with the new modem, then:

1. Log into the SPU as root
2. Using an editor, edit the file
/etc/ttytab
setting ttya to off.
3. Write file /etc/ttytab.

C3800 Troubleshooting Guide

4. Enter:

```
spu> kill -HUP 1
```

```
spu> /diag/bin/modem_init
```

5. Using an editor, edit the file

```
/etc/ttytab
```

6. Enter:

```
spu> kill -HUP 1
```

10.* C3800 SPU Console Window Messages

It should be understood that all messages occurring in the SPU Console window on the C3800 workstation do not indicate failures. Many of the, reported, messages are actually status messages and do not, necessarily, indicate a fault that needs user attention. Examples of status messages are as follow:

- 1) Scalar Halt message while executing substest 713 of spu4000.
- 2) Swip Bus Error after execution of cleanup and osclean from the SPU.
- 3) Intermittent "Framing Error" that indicate a missed transfer from the SWIS and a BPC. This transfer will then retry and correct the situation.
- 4) Overruns caused by a loss of transfer from the front panel key. Again the transfer is picked up successfully on the retry.

These "status" conditions were ignored in prior releases of the diagnostics and only after installing v1.0 have the conditions been reported. Because of the nature of these messages, it should always be assumed that there is no failure if everything seems to function properly.

11.* OPUS Status Utility "sfp"

There is a utility existing on the Opus that is used to find the status of the system and Opus. This utility is called "sfp" and stands for soft front panel.

With this utility it is possible to obtain the key switch position, the SPU printer status and the boot mode of the C3800. The display function is utilized by entering "sfp" at the spu prompt. This will generate the following return:

93/02/22
07:51:00

(trngps)(mikey:throg Job: vol.8 _ ate: Mon Feb 22 07:50:58 1993)

psfilter.in.23447

9

C3800 Troubleshooting Guide

```
switch position: LOCAL (or, remote, secure)
boot mode: diagnostic
printer: 0
```

The printer status is displayed as a 0, as above, or a 1. The 0 indicates off and the 1 indicates the printer is online. To change the status of the printer, remotely, it is necessary to enter `sfp -p x`, where `x` is a 1, or 0 depending on the desired state of the printer.

12.* OPUS Backup and Restore

Backup

The `/etc/backup` script looks at `/etc/fstab` to determine which file systems to dump.

Here is what `"fstab"` looks like on a SPU running SpuOS 2.0 and Diags 3.0.

```
/dev/sd0a / 4.2 rw 1 1
/dev/sd0f /diag 4.2 rw 1 3
/dev/sd0e /mnt 4.2 rw 1 4
/dev/sd0h /sst 4.2 rw 1 5
/dev/sd0g /usr 4.2 rw 1 2
```

`"/etc/backup"` will dump the file systems in the following order:

```
/
/diag
/mnt
/sst
/usr
```

Be sure to make note of the order the file systems are dumped, so you will have the proper info if you need to `"restore"` from the `"backup"` tape.

To backup the SPU do the following:

Insert a write enabled tape in the DAT drive.

```
spu> su to become root.
spu\> cd /
spu\> /etc/backup
```

This will dump all of the SPU file systems onto one tape.

Restore

Restoring `"/"`

At the `>b` prompt
Insert the `"SpuOS"` tape in the DAT drive.

```
>b st()
Probing Memory.....
Booting from: st(0,0,0)
```

C3800 Troubleshooting Guide

What would you like to do?

```
1 - install mini-root
2 - exit into single user shell
Enter a 1 or 2: 1
```

Do you want to format and/or label disk `"sd0"`?

```
1 - yes, run format
2 - no, continue loading miniroot
3 - no, exit into single user shell
Enter a 1, 2, or 3: 2
```

Problem with tape: what do you want to do?

```
1 - retry the tape "st0"
2 - use a different tape unit
3 - abandon miniroot install and enter single user shell
Enter a 1, 2, or 3: 1
```

Mini-root installation complete.

What would you like to do?

```
1 - reboot using just-installed miniroot
2 - exit into single user shell
Enter a 1 or 2: 1
```

You will see boot messages and then the miniroot prompt `"#"`.

```
# newfs /dev/rsd0a
# mount /dev/sd0a /a
# cd /a
Insert the "backup" tape in the DAT drive.
# mt -f /dev/rst0 rew
# restore xf /dev/rst0

Specify next volume#: 1
# Set directory mode, owner, and times.
set owner/mode for "."? [yn] y
# cd /
# umount /a
#/etc/fscck /dev/rsd0a
# mount /dev/sd0a /a
# cd /usr/kvm/mdec
# installboot /a/boot bootsd /dev/rsd0a
# /etc/reboot
```

Restoring `"/usr"`

At the `>b` prompt
Insert the `"SpuOS"` tape in the DAT drive.

```
>b st()
Probing Memory.....
Booting from: st(0,0,0)
```

93/02/22
07:51:00

(trngps)(mikey:throg Job: vol.8 Date: Mon Feb 22 07:50:58 1993)
psfilter.in.23447

10

C3800 Troubleshooting Guide

```
What would you like to do?
1 - install mini-root"
2 - exit into single user shell"
Enter a 1 or 2: 1
.
Do you want to format and/or label disk "sd0"?
1 - yes, run format
2 - no, continue loading miniroot
3 - no, exit into single user shell

Enter a 1, 2, or 3: 2
.
Problem with tape: what do you want to do?
1 - retry the tape "st0"
2 - use a different tape unit
3 - abandon miniroot install and enter single user shell
Enter a 1, 2, or 3: 1
..
Mini-root installation complete.

What would you like to do?
1 - reboot using just-installed miniroot
2 - exit into single user shell

Enter a 1 or 2: 1
.
You will see boot messages and then the miniroot prompt "#".
.
# newfs /dev/rsd0g
# mount /dev/sd0a /a
# mount /dev/sd0g /a/usr
# cd /a/usr

Insert the "backup" tape in the DAT drive.

# mt -f /dev/rst0 rew
# restore xfs /dev/rst0 5
.
Specify next volume#: 1
# Set directory mode, owner, and times.
set owner/mode for "."? [yn] y
# cd /
# umount /a/usr
# umount /a
# /etc/fsck /dev/rsd0g
# /etc/reboot

Restoring "/diag"

Boot the system to single user.
```

C3800 Troubleshooting Guide

```
>b -s
# /etc/newfs /dev/rsd0f
# mount /dev/sd0f /diag
# cd /diag

Insert the "backup" tape in the DAT drive.

# mt -f /dev/rst0 rew
# restore xfs /dev/rst0 2
.
Specify next volume#: 1
set owner/mode for "."? [yn] y
# cd /
# umount /diag
# /etc/fsck /dev/rsd0f
# ^d (Go to multi-user)

Restoring "/mnt"

Boot the system to single user.

>b -s
# /etc/newfs /dev/rsd0e
# mount /dev/sd0e /mnt
# cd /mnt

Insert the "backup" tape in the DAT drive.

# mt -f /dev/rst0 rew
# restore xfs /dev/rst0 3
.
Specify next volume#: 1
set owner/mode for "."? [yn] y
# cd /
# umount /mnt
# /etc/fsck /dev/rsd0e
# ^d (Go to multi-user)

Restoring "/sst"

Boot the system to single user.
>b -s
# /etc/newfs /dev/rsd0h
# mount /dev/sd0h /sst
# cd /sst

Insert the "backup" tape in the DAT drive.

# mt -f /dev/rst0 rew
# restore xfs /dev/rst0 4
.
```

93/02/22
07:51:00

(trngps)(mikey:throg Job: vol.8 Date: Mon Feb 22 07:50:58 1993)

psfilter.in.23447

11

C3800 Troubleshooting Guide

```
Specify next volume#: 1
set owner/mode for ".?" [yn] y
# cd /
# umount /sst
# /etc/fsck /dev/rsd0h
# ^d (Go to multi-user)
```

13.* Defective C3800 Keyswitches

It has been discovered that some defective keyswitches exist on installed C3800's. The keyswitches are defective in different positions and can cause individual bays to power down when moved from one position to another. This in turn will cause the system to crash.

If this problem is encountered, it is recommended that the keyswitch be replaced. The part number is 500-000416-202.

14.* Installing C3800 SpuOS 2.0

The install procedures in appendix A of the Convex SpuOS v2.0 release notes needs some corrections.

Appendix A step 4 should read:

```
Step 4: tar cvf /dev/rst0 /ioconfig /etc/hosts /etc/passwd /mnt (and any
other files you have modified)
```

Appendix A step 7 thru 9 should read:

```
Step 7: su (to become root)
Step 8: cd /
Step 9: tar xpf /dev/rst0
```

15.* Auto Reboot Script Warning

It should be understood that when selecting the network option during installation of the auto reboot daemon, on 3800's, the system is exposed to potential random reboots caused by network related failures.

The failures will generally appear as a system reboot for no apparent reason, which after much investigation, generally leads to a network fault.

Because of this possible problem, it is recommended that the response to the network option in the install script, be answered no. The specific question is:

Do you want ADB to ping \$host to see if it is up (yes or no)?

The default of ABD is to monitor the processes over the SWIP/NCU link. This is sufficient to reboot on hangs and crashes.

C3800 Troubleshooting Guide

16.* Spu Prompts

There are 2 spu prompts utilized in the C3800 CONVEX_OS window.

There is the standard "spu>" type prompt which indicates that OS is not running on the system and is generally in dshell.

The other prompt, "spu[2]#convexos!" indicates that the system is in OS and a ^D will return you to the system OS level. This prompt will change to the standard spu> prompt as soon as the OS is halted; either by a shutdown, or an osclean.

Before attempting a boot to the OS a check of this prompt should be accomplished.

It is recommended that this hint be passed to operators to prevent confusion.

17.* SPU Software Restore

Please, keep in mind, when restoring software on the C3800 SPU it is necessary to do so from /. It is very easy to make the mistake of doing a restore from diaguser, as that is where the work station boots to.

If the restore is not done from /, then the user id's will be changed and it will not be possible to boot, or worse.

18.* SPU Utility Execution

As a reminder, it should be understood that nearly all of the standard C3800 SPU level utilities are designed to function under dshell. These utilities will not function if executed from a non-dshell window.

In addition to set paths properly, they should be executed under diaguser.

If attempting to execute these utilities under su, or root, it should be realized that root on the SPU uses bourne shell. This means that these commands and utilities will not function.

Some examples of commands, or utilities that will be affected are the get/put, rslog, xbar_err, nmb_errs, etc.

19.* XSFP Key-Switch Positions and Power-up Actions

1 Key Off

1.1 SPU and 3800 powered down. Power-up Boot Mode doesn't matter. When the SPU boots, it will start the window manager and open the SPU CONSOLE, Xterm, XSFP and the Icons windows. It will open the CONVEXOS CONSOLE window and then iconify it. The SPU will then wait for further instructions.

93/02/22
07:51:00

(trngps)(mikey:throg Job: vol.8 Date: Mon Feb 22 07:50:58 1993)

psfilter.in.23447

12

C3800 Troubleshooting Guide

- 1.2 SPU and 3800 powered up and then key to off.
With old style keyswitch, will force 3800 power system into reset and immediately power down the 3800.
With new style switch, will call "sys_shutdown" utility and do a controlled power down of the 3800.
- 2 Key Local
 - 2.1 When the key is in the local position and the SPU is booted, the Xwindow pointer may be moved to any position on the screen. The user "rmtdiag" is not allowed to login.
 - 2.2 SPU and 3800 powered down. Power-up Boot Mode in "diagnostics". When the SPU boots, it will start the window manager and open the SPU CONSOLE, Xterm, XSFP, CONVEXOS CONSOLE and the Icons windows. The SPU then waits for further instructions.
 - 2.3 SPU and 3800 powered down. Power-up Boot Mode in "normal OS". When the SPU boots, it will start the window manager and open the SPU CONSOLE, Xterm, XSFP, CONVEXOS CONSOLE and the Icons windows. The SPU will then execute "diaginit" which will power-up the 3800. After the "diaginit" is complete, the SPU will execute a "boot multi" command and boot the 3800.
 - 2.4 SPU and 3800 powered down. Power-up Boot Mode in "alternate OS" When the SPU boots, it will start the window manager and open the SPU CONSOLE, Xterm, XSFP, CONVEXOS CONSOLE and the Icons windows. The SPU will then execute "diaginit" which will power-up the 3800. The SPU then waits for further instructions.
- 3 Key Remote
 - 3.1 When the key is in the remote position and the SPU is booted, the Xwindow pointer may be moved to any position on the screen. The user "rmtdiag" is allowed to log in.
 - 3.2 SPU and 3800 powered down. Power-up Boot Mode in "diagnostics". Functions the same as when key is in local.
 - 3.3 SPU and 3800 powered down. Power-up Boot Mode in "normal OS". Functions the same as when key is in local.
 - 3.4 SPU and 3800 powered down. Power-up Boot Mode in "alternate OS" Functions the same as when key is in local.

C3800 Troubleshooting Guide

- 4 Key Secure
 - 4.1 When the key is in the secure position and the SPU is booted, The Xwindow pointer is locked into the CONVEXOS CONSOLE window. The ^p and ^d functions are disabled. The modem's "getty" is killed and the ethernet is shutdown.
 - 4.2 SPU and 3800 powered down. Power-up Boot Mode in "diagnostics". Functions the same as when key is in local.
 - 4.3 SPU and 3800 powered down. Power-up Boot Mode in "normal OS". Functions the same as when key is in local.
 - 4.4 SPU and 3800 powered down. Power-up Boot Mode in "alternate OS" Functions the same as when key is in local.

20.* Spuio Problem

A problem with the spuio link between the 3800 and the spu is, currently, being experienced. The failure manifests itself as a spu I/O error at the CONVEXOS level when attempting to accomplish spu commands. Quite often, though not always, this will be accompanied by a swip bus error.

The cause of the failure is not known, at this time, but customers should be discouraged from initiating heavy activity with this link.

In the mean time, it is requested that the spuio process be killed and restarted at the SPU level. This can be accomplished by the following command sequence.

```
kill -9 pid  
spuio &
```

It is not certain that this will restore the link, but should be attempted.

This only affects the link from the 3800 to the SPU and does not impact the link from the SPU to the system. This failure will result in the failure of the APR mechanism and will eventually lead to system hangs, if not detected and fixed.

21.* Spu Tape boot failure

A problem has been discovered when booting SpuOS 2.0 from tape directly after power up. This problem should only arise when installing a new unloaded disk in a 3800 spu.

The solution is to install SpuOS 1.0 first.

93/02/22
07:51:00

(trngps)(mikey:throg Job: vol.8 Date: Mon Feb 22 07:50:58 1993)
psfilter.in.23447

13

C3800 Troubleshooting Guide

To work around this problem, do the following:

1. Insert a SpuOS 1.0 tape in the DAT drive.
2. At the ">" prompt type "n" <cr>.
3. At the "ok" prompt type "boot st()" <cr>.
4. At the "Boot:" prompt type "st(,,4)" <cr>.
5. When you see the "Initialize ram disk from device:" question, press the "L1" and "a" key at the same time.
6. Insert the SpuOS 2.0 tape in the DAT drive.
7. At the ">" prompt type "b st()" <cr>.

You may then install SpuOS 2.0 according to the "C3800 Service Processor Unit Service Guide" 2nd Edition chapter 4.2.1 or restore SpuOS 2.0 according to Tech Bulletin Vol. 4 Week 1 Article 5.

C3800 Troubleshooting Guide

CHAPTER II (DIAGNOSTICS)

1.* C3800 PETS PROBLEMS/HINTS

When executing PETS on a C3 system it is possible to encounter a Bus Error while attempting to execute run96. This can occur because a defective csh is shipped on the system. This problem can be remedied by going to single user and saving the file /bin/csh and copying /bin/oldcsh to /bin/csh.

This bus error, or coredump can happen on any subtest and will usually occur after execution for 30 minutes to an hour.

The PETS executed on a C3800 is not the same as that previously used on the C2.

The C3800 version is in a state of flux and is in the process of being ecn'd, so it can be distributed to the field. In the mean time each tape must be generated individually and must be ordered thru the TAC.

The C3800 version of PETS is not in installsw format, but is currently only in tar format.

Version 3.0 of PETS has been released which is in installsw format and usable on all C series systems.

Loading PETS through the SPU on a C3800

The facility for loading PETS through the SPU can be accomplished, but is cumbersome.

If the DAT tape is made in Dallas, then it will arrive compressed. This will save some time, but the DAT tape can be created in the field. I will detail both procedures below.

When the DAT tape has been created for this purpose, then there will only be a single file on the tape. This will generally be labeled pets.tar. This tape should be loaded in the sst partition with the command tar xv. The /sst partition should be used because this is the only one large enough. After loading this file it is then possible to spu -r this file to the OS level and extract the contents of the file by executing "tar xvf pets.tar". This will create the PETS directory exactly as if it had been loaded in a conventional manner.

If this pre-compressed tape is not available the following procedure can be followed to get PETS to the OS.

- 1) Copy PETS to the DAT tape by tar
- 2) Copy DAT contents to /sst by "tar xv
- 3) create pets file by " tar cvf pets.tar PETS". This will create the file /sst/pets.tar.
- 4) Boot OS and from the mnt directory execute "spu -r sst/pets.tar".
- 5) Breakout PETS from the file by "tar xvf pets.tar".

C3800 Troubleshooting Guide

This procedure will accomplish loading PETS so that it can be executed.

PETS Startup Problem

It is possible that after installing PETS that the scripts will appear to start, but no load will be achieved. This is usually because PETS has been loaded under different directory than was intended. The directory used is /mnt, but tapes in installsw format will prompt for the necessary directory during install and thus make path name modifications to the scripts. In cases where PETS is installed in tar format, such as C3800's, it will be necessary to load the tape under /mnt, or modify the run scripts with the different path name.

PETS Version 3.0

This is a combined list of known 3.0 related PETS failures. Although many of these appear in other technical bulletins, a comprehensive list of failures should be of benefit:

- 1) A bug causing wrong answers in prll.009 due to a missing expected answer file. This bug will result in WRONGANS failures with this subtest. (C34XX specific, only evident when running systems script)
- 2) Screen has a bug caused by an invalid margin sequence for cl. This should be corrected from "u ul" to "uu ul". (Cl Specific)
- 3) In the script "run48" the goto statement in the C3400 switch statement has a ":". This ":" should be at the actual goto statement for TypeParallel. If not corrected, this will cause the script to exit. (34XX specific)
- 4) Test 14 and 15 will now indicate that they will not execute. In previous releases these tests would indicate that they were run once and never repeat. The tests are disabled due to conflicts with file system frag sizes greater than 1K.
- 5) On C3800 systems it will be necessary to comment test.046 out of the scripts and move the module, in the Processes directory, to test046.sav. This is necessary because this test module is defective on the released version. Not removing test.046 from the scripts will result in a core dump.
- 6) The test.041 is moved for C3800's for much the same reason.
- 7) All part1 tests were broken by the new mpa in ConvexOS 10.1. This can be solved by loading the previous mpa utility in PETS/bin and moving PETS/bin before \$path in path+aliases.
- 8) On all systems, it is possible to encounter failures with the run96 script due to the incorrect /bin/csh being installed on the system. The failures exhibited can range from core dumps to the script appearing to start, but cease running after 30-45 minutes. In the latter case, no error indication will be given, but PETS will just die. To get around this problem it is necessary to copy /bin/oldcsh to /bin/csh at single user and return to multi.

C3800 Troubleshooting Guide

- 9) On C3800's running APR, if a head is removed from the complex, while executing PETS part1, then all subtest will result in coredumps. In addition, when the head is returned to the complex, no tests will be executed on this head until PETS is restarted.

2.* C3800 Continuity Test After Board Replacement

It should be understood that anytime a board is replaced on installed in the C3800 that it is necessary to run the appropriate continuity tests, as it is possible that the board has not seated properly. The continuity tests necessary will normally involve spu4000 subtest 810. Which will verify connectivity for all installed hardware in the system. The utilities xc_con and xbindeg have been merged into subtest 810, but are still available for individual crossbar related tests.

The discussion of the various continuity tests is detailed in topic #3 in this chapter. Please read this for further details.

If the continuity tests are not executed, it is possible that failures in OS can be misinterpreted.

3.* C3800 pram.log Created by New hard_logger Dumpers

Listed below is the contents of the pram.log created by the new hard_logger dumpers on the C3800. This includes the error output from all 19 purge rams. This list includes their names and an error.

As can be seen the error resides in the purge ram ic_valid at location 7b0. The data (1) the data and address validity differ. This should not occur, so the only valid values in these 2 bit registers is a 0, or a 3, indicating the data and address are either valid or invalid. The error occurs when the address and data entries disagree with one another.

The error can be verified by checking the location indicated in the affected ram. A collection of all ram outputs is created in the file pram.stdout. To locate the failure, simply search on the ram name and then the address in the file pram.stdout.

Included with the pram.log is a comment for each ram indicating actual function. These definitions are not included in the actual dump.

Ram : la_valid	Look Ahead Validity
Ram : ic_valid	Instruction Cache Validity
7b0 1	
Ram : br_hist_l	Branch History Cache read data lower word
Ram : br_hist_xl	Branch History Cache read data lower word
Ram : br_hist_u	Branch History Cache read data upper word
Ram : br_hist_xu	Branch History Cache read data upper word
Ram : pte_mod1	PTE Cache odd modified bits
Ram : pte_refl	PTE Cache odd referenced bits
Ram : pte_tvall	PTE cache odd thread validity
Ram : pte_vall	PTE cache odd validity

C3800 Troubleshooting Guide

```
Ram : pte_mod0      PTE Cache even modified
Ram : pte_ref0      PTE Cache even referenced
Ram : pte_tval0     PTE Cache even thread validity
Ram : pte_val0      PTE Cache even validity
Ram : dc_tvall      Data Cache odd thread validity
Ram : dc_vall       Data Cache odd validity
Ram : dc_tval0      Data Cache even thread validity
Ram : dc_val0       Data Cache even validity
Ram : sr_val        Scratch Ram validity
```

4.* C3800 Diagnostic Bugs

The following are diagnostic bugs observed with Systems Diagnostics V1.0.

1. cpu4332 - At 16 ns this diagnostic will fail. If you put the osc_freq to 18 ns, this test may be used to help isolate a fault.

2. ncutestall - D4 of this script fails with the following message.

```
FAILED ncutestD4
Fails TRPCMD lock testing.
Check DLCK CNT2, L2_LCKLCKB, L3_TRPLCKB, L2_TRPLCKB, TRP_BSY, ITC_INT,
and CTL_TRP_LCKB.
```

3. xbinteg e - Fails with the following message.

```
TESTING - SENDSEL = 0
ERROR: xrte:rctl.rm1_rdy = 0x0 (expect 0x1)
ERROR: xrte:rctl.rm2_rdy = 0x0 (expect 0x1)
ERROR: xrte:rctl.rm3_rdy = 0x0 (expect 0x1)
```

4. xbinteg o - Fails with the following message.

```
TESTING - SENDSEL = 0
ERROR: xrto:rctl.rm1_rdy = 0x0 (expect 0x1)
ERROR: xrto:rctl.rm2_rdy = 0x0 (expect 0x1)
ERROR: xrto:rctl.rm3_rdy = 0x0 (expect 0x1)
```

5.* Verifying C3800 Control Stores

The proper procedure for verifying control stores on a C3800 is to execute "sysreset -l 2 -v". This will load and verify all system control stores on each head.

6.* Multi-head Troubleshooting Utilities

As most everyone knows, there are a couple of utilities available on all CONVEX multi headed systems that can be very useful in troubleshooting.

These utilities are mpa and cpuconf.

C3800 Troubleshooting Guide

They can be especially useful in chasing application failures, wrong answer problems or most any other type of problem where the system does not crash and could be individual head dependent. Both utilities have the convenience of functioning at the OS level and so preventing lost time with reboots to use the spu disable utility. In addition, there are situations where disabling the head at the spu and thus not loading microcode can actually lead to false conclusions.

The utility, mpa, is useful in locking a process on to an individual head. In this way the heads can be isolated with the particular application one at a time and very quickly. You can even have the user, or operator perform this function. The format of this utility is as follows:

```
mpa -c x application script <where x is the processor from 0-7>
an example would be "mpa -c 1 run < input file"
```

The utility, cpuconf, performs opposite to mpa. This will allow you to remove a head from scheduling and thus run an application on all heads except one. This can be helpful in isolating parallel application failures. The format of this utility is as follows:

```
<prompt> cpuconf -d 0
<prompt> run < input file > output file
<prompt> cpuconf -e 0 -d 1
```

where -d switch is used to disable the head on line and -e is used to enable a head. As can be seen, it is permissible to operate both functions on any number of heads simultaneously.

Use of these utilities can save many hours of unnecessary labor and also prevent unwanted customer down time. These utilities can be run while the customer is using the system.

7.* C3800 spu4000 connectivity test (signals not covered in current release)

This does not include any scan or clock related signals in the list.

Memory Board <-> Xbar	Tested Where:
mb_xc.soft_error	spu4000 st_710-713
mb_xc.hard_error	spu4000 st_710-713
mb_xc.ram_rfsh	mem4000

Scalar Processor <-> Xbar

xc_sp.trap_type<0-3>	
xc_sp.trap_vec<0-11>	
xc_sp.usec_en	
xc_sp.deadlock	
xc_sp.cu_status_en	
xc_sp.cu_status	
xc_sp.mt_comp	
xc_sp.trap_rdy	
sp_xc.stop_cntr	spu4000 st_710-713

93/02/22
07:51:00

(trngps)(mikey:throg Job: vol.8 Date: Mon Feb 22 07:50:58 1993)

psfilter.in.23447

16

C3800 Troubleshooting Guide

Vector Processor <-> Xbar
vp_xc.hard_error spu4000 st_710-713

NCU <-> Xbar

xc_cu.trap_type<0-3>
xc_cu.trap_vec<0-11>
xc_cu.trap_rdy<0-8>
xc_cu.status_psel<0-3>
xc_cu.status_en
xc_cu.status
xc_cu.deadlock<0-7>
xc_cu.trap_comp<0-8>
xc_cu.usec_en
cu_xc.usec_en
cu_xc.xcl_addr<0,2,3,4>
cu_xc.xcl_addr_par
cu_xc.xcl_mux_ct1<0-2>

NIA <-> Xbar

ia_cu.trap_vec<0-11>
ia_xc.hard_errorspu4000 st_710-713
ia_xc.soft_errorspu4000 st_710-713
xc_ia.status_en
xc_ia.status

8. C3800 Net Resistance Value

It should be noted that any data net on the C3800 that exceeds 56 ohms should be considered a possible source of failure. Because of the system edge rates, any net that is not double terminated that is above this value can cause intermittent failures.

Causes of nets exceeding this value are normally epont, or augat related. These nets can cause very intermittent failures and can be difficult to locate.

When measuring point to point from heads to crossbar, these nets should measure less than 2.5 ohms. Any net exceeding this value should be viewed suspiciously.

Epont cables should measure less than 1 ohm point to point, with the same cautions as above.

WARNING Verify meter is not in auto-ranging, as this can cause erroneous readings.

9.* Continuity Test Failure Before sysreset

It is not possible to successfully complete any continuity tests, such as sys_con, or xc_con, before a sysreset has been accomplished. All failures from continuity tests should be ignored until after a sysreset has been accomplished.

C3800 Troubleshooting Guide

It is also not possible to complete diagnostics until an initial has been accomplished. In most cases, until the initial is complete the diagnostics will abort with a bus error. But, in some cases the diagnostic will fail after running for an extended period. This is especially true for mem4000.

10.* IA Soft Errors and rslog Utility

The following is a display for the utility rslog. This utility can be very helpful in locating the source of IA soft errors and mminit failures, found on C3800's.

This utility can be executed by entering "rslog" at the SPU prompt.

port	IA8 soft log ring										ll	wrt	rd	wrt	hdr	ccu0	ccu1	ccu
	7	6	5	4	3	2	1	0	hdr	pcm								
nx1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0
pbi0	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	1
pbi1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	1
pbi2	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	1
pbi3	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	1

11.* Remote Execution of diags spu4000, cu4000 and mem4000

As ddb cannot be utilized to activate spu tests (spu4000, cu4000 and mem4000, you must utilize dsh if you want to run these tests from a remote location.

The following are some examples of dsh test methods. This is not intended to cover all situations but should provide enough to cover most. All entries after the xxx> prompts are entered by you.

1. An example of running cu4000.

```

spu> cu4000

+++>
<Thu Mar 26 12:41:45 1992> cu4000:../../cti.c:242
Test Start (DiagER299): Test Startup Event

test name: cu4000
cwd: /users/diaguser
cmdline: cu4000
****
STARTUP> sub_all *(This will ensure that all subtests are run)*
subtest class description timeout dep
100 1 Communication Register Parity Error Generation. 45
105 2 Communication Register Lock Bit Test. 1000
110 2 Communication Register Pattern Test. 1000
115 2 CU Control Space Test. 45
120 3 Communication Register Functionality Test. 300

```

93/02/22
07:51:00

(trngps)(mikey:throg Job: vol.8 Date: Mon Feb 22 07:50:58 1993)
psfilter.in.23447

17

C3800 Troubleshooting Guide

```
200 3 CU ASAP Logic Testing. 45
210 3 CU RDCMR/WRCMR Testing. 45
220 3 CU TOC and ITC Functionality Testing. 45
230 3 CU Deadlock and Firmware Traps Testing. 45
300 3 CU Interrupt Test. 120
```

```
STARTUP> run *(Starts the test)*
info: "cu4000 subtest 100 running"
```

```
+++>
<Thu Mar 26 12:42:25 1992> cu4000:../ncu4000.c:137
SW Info (DiagIN230): Test complete
```

```
Test: cu4000 R1.0 Subtest: 100 R1.0 Class: 1
Purpose: Communication Register Parity Error Generation.
```

```
***** SUBTEST SUCCESSFULLY COMPLETED *****
****
```

```
info: "cu4000 subtest 105 running"
***** Now testing lock bit 0000 *****
***** Now testing lock bit 0100 *****
***** Now testing lock bit 0200 *****
***** Now testing lock bit 0300 *****
***** Now testing lock bit 0400 *****
```

```
^CPRE-CLEANUP> exit *(At this point the test was stopped with a ^C
then did an exit)*
```

```
+++>
<Thu Mar 26 12:44:05 1992> cu4000:../CTIsequencer.c:469
Test End (DiagER316): CTI Test Summary
```

```
test name: cu4000
runtime: 1:26.246
subtests started: 2
subtests passed: 1
subtests failed: 0
****
```

```
+++>
<Thu Mar 26 12:44:05 1992> cu4000:../CTIutil.c:229
Test End (DiagER300): Test Termination Event
```

```
test name: cu4000
status: PASSED
****
```

2. An example of running selected subtests of mem4000.

```
spu> mem4000
```

```
+++>
<Thu Mar 26 12:50:19 1992> mem4000:../cti.c:242
Test Start (DiagER299): Test Startup Event
test name: mem4000
cwd: /users/diaguser
cmdline: mem4000
```

C3800 Troubleshooting Guide

```
****
ERROR "CPU 0 is not available for testing"
ERROR "CPU 1 is not available for testing"
ERROR "CPU 2 is not available for testing"
ERROR "CPU 3 is not available for testing"
ERROR "CPU 4 is not available for testing"
ERROR "CPU 5 is not available for testing"
ERROR "MB 0 is not available for testing"
ERROR "MB 1 is not available for testing"
ERROR "MB 2 is not available for testing"
ERROR "MB 3 is not available for testing"
ERROR "MB 4 is not available for testing"
ERROR "MB 5 is not available for testing"
STARTUP> sequence 380-400 *(To run only selected subtests)*
STARTUP> run
info: "mem4000 subtest 380 running"
```

```
+++>
<Thu Mar 26 12:52:30 1992> mem4000:../error_log.c:266
SW Info (DiagIN230): Test complete
```

```
Test: mem4000 R1.0 Subtest: 380 R1.0 Class: 4
Purpose: Memory testing via Memory Test Logic: address pattern.
```

```
***** SUBTEST SUCCESSFULLY COMPLETED *****
****
```

```
info: "mem4000 subtest 385 running"
Testing with pattern = 00000000 00000000, op = HOLD
Testing with pattern = ffffffff ffffffff, op = HOLD
Testing with pattern = a5a5a5a5 a5a5a5a5, op = HOLD
Testing with pattern = 5a5a5a5a 5a5a5a5a, op = HOLD
Testing with pattern = 69696969 69696969, op = HOLD
Testing with pattern = 96969696 96969696, op = HOLD
Testing with pattern = c3c3c3c3 c3c3c3c3, op = HOLD
Testing with pattern = 3c3c3c3c 3c3c3c3c, op = HOLD
Testing with pattern = 00000001 00000001, op = LEFT SHIFT
Testing with pattern = ffffffff ffffffff, op = LEFT SHIFT
Testing with pattern = 11111111 11111111, op = LEFT SHIFT
Testing with pattern = eeeeeeee eeeeeeee, op = LEFT SHIFT
Testing with pattern = ffffffff ffffffff, op = COMPLEMENT
Testing with pattern = a5a5a5a5 a5a5a5a5, op = COMPLEMENT
Testing with pattern = c3c3c3c3 c3c3c3c3, op = COMPLEMENT
Testing with pattern = 96969696 96969696, op = COMPLEMENT
Testing with pattern = 0fffffff 0fffffff, op = HOLD
```

```
+++>
<Thu Mar 26 12:54:13 1992> mem4000:../error_log.c:266
SW Info (DiagIN230): Test complete
```

```
Test: mem4000 R1.0 Subtest: 385 R1.0 Class: 4
Purpose: Memory testing via Memory Test Logic: various patterns.
```

93/02/22
07:51:00

(trngps)(mikey:throg Job: vol.8 Date: Mon Feb 22 07:50:58 1993)

psfilter.in.23447

18

C3800 Troubleshooting Guide

***** SUBTEST SUCCESSFULLY COMPLETED *****

```
****
info: "mem4000 subtest 400 running"
Testing with pattern = 00000000 ffffffff
Testing with pattern = ffffffff 00000000
Testing with pattern = 55555555 00000028
Testing with pattern = 0bbbbbbb cccccccc
Testing with pattern = 33333333 0f0f0f0f
^CPRE-CLEANUP> exit
```

```
+++>
<Thu Mar 26 12:56:01 1992> mem4000:.././CTIsequencer.c:469
Test End (DiagER316): CTI Test Summary
```

```
test name: mem4000
runtime: 3:22.238
subtests started: 3
subtests passed: 2
subtests failed: 0
```

```
+++>
<Thu Mar 26 12:56:02 1992> mem4000:.././CTIutil.c:229
Test End (DiagER300): Test Termination Event
```

```
test name: mem4000
status: PASSED
****
spu>
```

3. An example of spu4000, st 810 only

spu> spu4000

```
+++>
<Thu Mar 26 12:35:48 1992> spu4000:.././cti.c:242
Test Start (DiagER299): Test Startup Event
```

```
test name: spu4000
cwd: /users/diaguser
cmdline: spu4000
STARTUP> com *(Displays available commands)*
- - "!" "passes a quoted string to the shell"
- - "abort" "aborts a subtest if in "interrupt" or "run" state,
otherwise aborts the test altogether"
- - "base" "sets the default output radix - 2, 8, 10, or 16"
- - "changed" "[filename] print the changed parameters and their values in
order changed"

- - "class" "[class]
omit class to see all classes
supply class to see info on that class and all of its subtests"
- - "com" "[name] show the help string for a command or parameter"
- - "cont" "runs the subtest sequencer"
- - "dep" "[first-last] show the subtests, omit the range for all"
```

C3800 Troubleshooting Guide

```
- - "exit" "cleanup, then exit"
- - "get_par" "read in a set of parameters from the supplied filename"
- - "globals" "show parameters that have the CTI_GLOBALS flag set"
- - "help" "tells how to find information"
- - "log_file" "specifies a file for message logging"
- - "map" "map the specified window to the ctix display"
- - "par" "[par_name] show the parameters and their values"
- - "quit" "cleanup, then exit"
- - "rerun" "runs the sequencer starting with previous subtest"
- - "reset" "resets the CTI, sequencer, counters, & status"
- - "run" "specify a subtest sequence and restart sequencer"
- - "save" "[filename] save changed parameter values to the file."
- - "save_all" "[filename] save all parameter values to the file."
- - "skip" "skip n subtests in the sequence"
- - "status" "request a status update"
- - "sub" "[first-last] show the subtests in the run list, omit the range
for all"
- - "sub_all" "[first-last] show the known subtests, omit the range for all"
- - "summary" "write parameters that have the CTI_SUMMARY flag set."
- - "test_par" "show parameters that are specific to the test"
- - "update_selects" "Select each installed board"
```

```
STARTUP> sequence 810 (To startup on st_810)*****
STARTUP> update_selects (Ensures all installed boards are tested)*****
STARTUP> run
```

```
info: "spu4000 subtest 810 running"
Running Scalar Processor 6 to Crossbar Connectivity Test.
Running Scalar Processor 7 to Crossbar Connectivity Test.
Running Crossbar to Scalar Processor 6 Connectivity Test.
Running Crossbar to Scalar Processor 7 Connectivity Test.
Running Scalar/Vector Processor 6 Connectivity Test.
Running Scalar/Vector Processor 7 Connectivity Test.
Running Vector/Scalar Processor 6 Connectivity Test.
Running Vector/Scalar Processor 7 Connectivity Test.
Running Memory Board 6 to Crossbar Connectivity Test.
Running Memory Board 7 to Crossbar Connectivity Test.
Running Crossbar to Memory Board 6 Connectivity Test.
Running Crossbar to Memory Board 7 Connectivity Test.
Running Interface Adapter 8 to Crossbar Connectivity Test.
Running Crossbar to Interface Adapter 8 Connectivity Test.
Running NCU to Crossbar Connectivity Test.
Running Crossbar to NCU Connectivity Test.
```

```
+++>
<Thu Mar 26 12:39:14 1992> spu4000:.././CTIsequencer.c:469
Test End (DiagER316): CTI Test Summary
```

```
test name: spu4000
runtime: 1:19.035
subtests started: 1
subtests passed: 1
subtests failed: 0
****
```

COMPLETE> exit

```
+++>
<Thu Mar 26 12:40:30 1992> spu4000:.././CTIutil.c:229
Test End (DiagER300): Test Termination Event
```

93/02/22
07:51:00

(trngps)(mikey:throg Job: vol.8 Date: Mon Feb 22 07:50:58 1993)

psfilter.in.23447

19

C3800 Troubleshooting Guide

```
test name: spu4000
status: PASSED
****
spu> spu4000
```

```
+++
<Thu Mar 26 12:40:48 1992> spu4000:.././cti.c:242
Test Start (DiagER299): Test Startup Event
```

```
test name: spu4000
cwd: /users/diaguser
cmdline: spu4000
****
```

When running spu4000, on a C3800, in a remote situation, it is possible to display the test parameter's by way of the "par" command. This will display all of the system parameter's that can be displayed when running under the xdiag mode.

An example of the parameters can be seen in the example below:

```
test name: spu4000
cwd: /users/diaguser
cmdline: spu4000
****
STARTUP> par
```

```
- "CTI_path" "./"
- "CTI_pathname" "spu4000"
- "CUIlexists" "0"
- "alarm" "5"
- "cleanup_messages" "0"
- "complete_on_fail" "0"
- "confirm_abort" "0"
- "cu_selected" "0"
- "current_description" ""
- "current_subtest" "0"
- "event_logger" "1"
- "fail_messages" "1"
- "failure_threshold" "10"
- "force_fail" "0"
- "ia_selected" "0,0,0,0,0,0,0,0,0"
- "init_messages" "0"
- "log_event" "0"
- "log_input" "0"
- "log_output" "0"
- "loop_count" "1"
- "looping" "0"
- "mb_selected" "0,0,0,0,0,0,0,0"
- "next_description" ""
- "next_subtest" "410"
- "pass_messages" "0"
- "pattern_number" "0x00000000"
- "pause_before_subtest" "0"
- "pause_between_subtest" "0"
- "pause_on_fail" "0"
- "pause_on_pass" "0"
```

C3800 Troubleshooting Guide

```
- "pause_when_complete" "1"
- "prompt" ">"
- "ring_number" "40"
- "run_messages" "1"
- "runtime_messages" "0"
- "sequence" "(410-421), (510-511), (610-614), (710-713), (810)"
- "sp_selected" "0,0,0,0,0,0,0,0"
- "state" "1"
- "subtest_failures" "0"
- "subtest_passes" "0"
- "subtest_runtime" "0"
- "subtest_timeout" "0"
- "subtests_remaining" "23"
- "subtests_started" "0"
- "test_runtime" "0"
- "test_summary" "1"
- "timeout_multiplier" "1"
- "updates" "0"
- "vp_selected" "0,0,0,0,0,0,0,0"
- "where_paused" "0"
- "why_paused" "0"
- "xbar_selected" "1"
```

To display individual boards selected to test, the individual entry can be entered as seen below:

```
STARTUP> "mb_selected[0]" "0"

"mb_selected[1]" "0"
"mb_selected[2]" "0"
"mb_selected[3]" "0"
"mb_selected[4]" "0"
"mb_selected[5]" "0"
"mb_selected[6]" "0"
"mb_selected[7]" "0"

STARTUP> "sp_selected[0]" "0"

"sp_selected[1]" "0"
"sp_selected[2]" "0"
"sp_selected[3]" "0"
"sp_selected[4]" "0"
"sp_selected[5]" "0"
"sp_selected[6]" "0"
"sp_selected[7]" "0"
```

12.* Error Rules for CXTS

It has frequently been asked how to proceed with analysis of different failures. While this is a very difficult question to answer, there are some basic guidelines in existence from engineering. Below is listed rules that have been generated for incorporation into CXTS. These analysis rules are to aid in the decision process for determination of troubleshooting paths. The rules illustrated here are for the NCU, NIA, NSP and NVP.

93/02/22
07:51:00

(trngps)(mikey:throg Job: vol.8 Date: Mon Feb 22 07:50:58 1993)

psfilter.in.23447

20

C3800 Troubleshooting Guide

These rules are only guidelines and covered here for information purposes, so that Field personnel can become more familiar with the analysis and extractor processes.

Extractor Name.	Rule.
cu_lckb_par_err cu_ram_par_err	These error conditions are totally isolated to the CU and will not involve any other component of the system.
cu_trp_harderr	This error has been disabled by scan. Checked to verify the scan ring of the CU, by executing spu400 subtest 611. If this passes then there is some software that is not initializing the board correctly.
cu_wr_dat_par_err	This error is on bad write data from the XS10 bits <23..0>. If this error is seen then there should also be a cu_ndat0_harderr. IF there is a cu_ndat0_harderr then Check to see if there are any mb_iso_data_perr. IF there are, and there is bad data in the xbar and the memory board, the problem may involve the port identified by the MB. All receivers (in this case the CU is a receiver) could be detecting the error. Indicates a potential connectivity problem. Run cpu4332 from a processor that is not the one indicated by the mb_iso_data_perr. This will verify that there is basic connectivity at speed and spu4000 subtest 810 will probably not find anything. If this passes then run the test again from the processor that mb_iso_data_perr has identified. IF this passes, the problem could still be connectivity. It would be time for the meter. Check the value of the terminations at the CPU backplane for the three lower byte of the write data.
cu_ndat1_harderr	Check to see if there are any mb_ise_data_perr. IF there are, and there is bad data in the xbar and the memory board, the problem may be coming from the port which the MB has identified. All receivers (the CU is a receiver) could be detecting the error. Indicates a potential connectivity problem. Run cpu4332 from a processor that is not the one indicated by the mb_ise_data_perr. This will verify that there is basic connectivity at speed and spu4000 subtest 810 will probably not find anything. If this passes then run the test again from the processor that mb_ise_data_perr has identified. IF this passes, the problem could still be connectivity. It would be time for the meter. Check the value of the terminations at the CPU backplane for all the write data bits.
cu_ndat0_harderr	IF this error is received here is no cu_wr_dat_par_err THEN the problem is likely to be on the most significant byte of the data. (bits <31..24>). Use the same rules as above.

C3800 Troubleshooting Guide

Check to see if there are any mb_ise_data_perr. IF there are, and there is bad data in the xbar and the memory board, the problem may be coming from the port which the MB has identified.

- 1) Run cpu4332 from a processor that is not the one indicated by the mb_ise_data_perr. This will verify that there is basic connectivity to the CU at speed and spu4000 subtest 810 will probably not find anything.
- 2) IF this passes then run the test again from the processor that mb_ise_data_perr has identified.
- 3) IF this passes, the problem could still be connectivity. It would be time for the meter. Check the value of the terminations at the CPU backplane for the most significant byte of the write data.
- 4) IF cpu4332 fails at 1) then this failure would have to be diagnosed separately.
- 5) IF the failure was another cu_ndat0_harderr then the connectivity between the CU and the XBAR should be examined on bits <31..24> from the xbar at the IO backplane.
- 6) IF the connectivity is found to be good in 5) then switch the XS1 even and odd boards.
- 7) IF the problem stays on the same side then pick the NCU or the Augat to replace.

NOTE The ndat gate arrays do not follow conventional naming conventions, as ndat1 is even and ndat0 is odd.

cu_addr_par_err_3_2
cu_addr_par_err_1
cu_addr_par_err_0

The Rules for these will be the same. The difference is in the actual bits involved with detecting the error. The same rules apply to the address that is applied to the data. That is the cpu4332 test. If the data in the xbar does not match the data in the board then the problem is the connectivity at the CU - XBAR interface. If there were mb_iso_addr_perr then the processor is probably the source of the problem. If the diagnostics do run and without error the next thing would be to meter the bits on the byte that was flagged as the error. (_0, _1, _3_2). If these were found to be normal impedance (52-54 ohms) then the next step would be to switch the XS0 even and odd boards and see if the problem moved sides.

NIA rules

- # I - should be internal
- # F - should be internal, but might fall through from external
- # X - could be external
- # There are also some internals which might fall through to other internals,
- # but those cases are handled by hord_logger extractor rules.

93/02/22
07:51:00

(trngps)(mikey:throg Job: vol.8 _ate: Mon Feb 22 07:50:58 1993)

psfilter.in.23447

21

C3800 Troubleshooting Guide

```
# The notation 'if ( extractor )' is true when the extractor reports an error.
# The notation 'extractor.word' refers to a value passed by the extractor
# in the event report.
#
ia_xds_hdr_pe I XDS arrays detected a header parity error.
: reject NIA
ia_xds_wde_pe I XDS arrays detected even side write data parity error
on data from WDQ.
: reject NIA
ia_xds_wdo_pe I XDS arrays detected odd side write data parity error
on data from WDQ.
: reject NIA
ia_cds_rd_pe F CDS arrays detected read data parity error on data from RDQ.
: if ( ia_rdq_wrt_pe )
: { ignore this error }
: else
: { reject NIA }
ia_rdq_wrt_pe F Parity error detected on write data during write access
to RDQ.
: if ( ia_rdq_wrt_pe )
: { ignore this error }
: else
: { reject NIA }
ia_wdq_wrt_pe I Parity error detected on write data during write access
to WDQ.
: reject NIA
ia_pi_hard_error X Plus interrupt state machine hard error. CCU unexpectedly
dropped its interrupt request.
: /* start at 50% connection, 25% ccu, 25% nia */
: Ccu = ia_pi_hard_error.ccu
: if ( Ccu not installed ) { reject NIA }
: else
: {
: /* start at 50% connection, 25% ccu, 25% nia */
: run sst on nia
: run ccu_con on Ccu
: }
ia_nxi_hard_error X NXI interrupt state machine hard error. XIOP unexpectedly
dropped its interrupt request.
: Xiop = ia_nxi_hard_error.xiop
: if ( Xiop > 1 ) { reject NIA } /* bad error report */
: else if ( Xiop not installed ) { reject NIA }
: /* ia8 xiop0 is SPU */
: else
: {
: /* start at 50% connection, 25% xiop, 25% nia */
: run sst on nia
: run nxp.newcon on Xiop
: }
ia_wdq_fflag_pe I WDQ flush flag parity error.
: reject NIA
ia_rdq_flag_pe I RDQ read error flag parity error.
: reject NIA
ia_rd_par_err_e X Parity error on read data from even side crossbar.
: Port = ia_rd_par_err_e.port
```

C3800 Troubleshooting Guide

```
: if ( any other errors reported by even side xbar )
: { ignore this error }
: else if ( nia rtn_par_err stops xbar )
: /* if ( xrte:p_config & (1<<ia_rd_par_err_e.port) ) */
: {
: if ( ias$Port:r_rdata_e == xrte:look_aside_
: && ias$Port:r_rpar_e_sl_3_0 == xrte:look_aside_ )
: { }
: }
: else /* data not stored on xbar for comparison */
: {
: run sys_con for the NIA
: }
: }
ia_rd_par_err_o X Parity error on read data from odd side crossbar.
: Just like even side.
```

NSP rules

XBAR CASE

```
if extractor (nrc.xre_stop_in | nrc.xro_stop_in) then
if nrc.xre_stop_in then
check (xrt.rtn_par_err_sp? 0) for error
if there is a failure on the xrte, follow it's rules to
determine source of error.
if no error exists on xrte then failure is either
connectivity, a bad xrt, or a bad nsp
try spu4000 -s 812 (to test connectivity)
if no failure is found by spu4000 -s 812 then try
swapping xtre-xtro, and reboot looking for failure
to move, if it does then it's a bad xrt, if it
doesn't, then move the nsp between cpus and reboot
looking for the failure to move, if it does it's a
bad nsp, if not then there is a resistive open/short
not detectable by spu4000.
else if nrc.xro_stop_in then
check (xrt.rtn_par_err_sp? 1) for error
if there is a failure on the xrto, follow it's rules to
determine source of error.
if no error exists on xrto then failure is either
connectivity, a bad xrt, or a bad nsp
try spu4000 -s 812 (to test connectivity)
if no failure is found by spu4000 -s 812 then try
swapping xtre-xtro, and reboot looking for failure
to move, if it does then it's a bad xrt, if it
doesn't, then move the nsp between cpus and reboot
looking for the failure to move, if it does it's a
bad nsp, if not then there is a resistive open/short
not detectable by spu4000.
```

93/02/22
07:51:00

(trngps)(mikey:throg Job: vol.8 Date: Mon Feb 22 07:50:58 1993)

psfilter.in.23447

22

C3800 Troubleshooting Guide

note: in the case of a resistive open you will need to know what the data held on the xtre and the nsp are for each failure and check to determine if the same data/parity line is failing, in which case after board swaps are exhausted you can point to a data/parity net to check connectivity on.

```
if extractor (npsw.statq_hard_err) then
try spu4000 -s 812 and /diag/hw/cutest/xc_con p? (to test connectivity)
if no failure is found by either then you have a problem. :^)
This failure can be caused by 2 conditions, 1 is entirely contained
on the nsp, the other is caused by an extra pop of the cu-sp.stat_en
which actually goes from the ncu to the xcl and from the xcl to the
nsp. It could be any of the 3 boards or resistive open/short on either
net. I would try moving the processor and see if it follows it,
if it doesn't then try swapping either the xcl or ncu till the
problem goes away or both board have been swapped, in which case
it must be a resistive connectivity problem.
```

note: in most cases it is impossible to tell if the failure is the internal error or the ncu error, but in a select few cases it is, so we will have to decide on a way to communicate which if either it is.

```
if extractor (npsw.ucode_pulled_hard_err) then
this failure isn't really a hard error at all, it's more like a ConvexOS
fatal error pulled by the microcode on the nsp. In most cases the failure
is caused by one of the xrt boards stopping and bogus data being used
for instruction cache data. Check for hard errors in the xbar, if there
aren't any, then it could be just about any board in the system. Depending
on the uirl_upc value. If need be we can decide on more specific rules
but this is as far as I will go for now.
```

note: it may be nice if I pass the uirl_upc with this extractor, as it is about the only way to gain even a bit of insite into this failure.

NVP CASE

```
if extractor (nag0.vxaq_par_err | nag1.vxaq_par_err) then
check (vp?) for errors
if there is a hard error on the nvp, then ignore the
nsp errors for now unless the nvp rules end up pointing
at the nsp.

if no error exists on the nvp then failure is either
connectivity, a bad nvp, or a bad nsp
try spu4000 -s 812 (to test connectivity)
if no failure is found by spu4000 -s 812 then try
swapping nvp with another cpu (if possible),
and reboot looking for failure to move,
if it does then it's a bad nvp, if it
doesn't, then move the nsp between cpus and reboot
looking for the failure to move, if it does it's a
bad nsp, if not then there is a resistive open/short
not detectable by spu4000.
```

C3800 Troubleshooting Guide

If you only have 1 cpu then I would replace the nvp first and see if the failure goes away, if not replace the nvp and try the nsp. If the failure still persists then there is a resistive open/short not detectable by spu4000.

note: in the case of a resistive open you will need to know what the data held on the nsp was for each failure and check to determine if the same data byte is failing, in which case after board swaps are exhausted you can at least narrow the nets to be checked for connectivity to 9.

```
if extractor (ndp.yuvc_vx_err) then
check (vp?) for errors
if there is a hard error on the nvp, then ignore the
nsp errors for now unless the nvp rules end up pointing
at the nsp.
if no error exists on the nvp then failure is either
connectivity, a bad nvp, or a bad nsp
```

```
try spu4000 -s 812 (to test connectivity)
if no failure is found by spu4000 -s 812 then try
swapping nvp with another cpu (if possible),
and reboot looking for failure to move,
if it does then it's a bad nvp, if it
doesn't, then move the nsp between cpus and reboot
looking for the failure to move, if it does it's a
bad nsp, if not then there is a resistive open/short
not detectable by spu4000.
```

If you only have 1 cpu then I would replace the nvp first and see if the failure goes away, if not replace the nvp and try the nsp. If the failure still persists then there is a resistive open/short not detectable by spu4000.

note: in the case of a resistive open you will need to know what the data held on the nsp was for each failure and check to determine if the same data byte is failing, in which case after board swaps are exhausted you can at least narrow the nets to be checked for connectivity to 9.

ANY OTHER CASE

if any other extractor finds an error besides the ones listed above, then the error is entirely contained on the nsp and the board should be rejected.

NVP rules

NSP CASE

```
if extractor (nis0.is_par_err | nis1.is_par_err) then
check (sp?) for errors
if there is a hard error on the nsp, then ignore the
nvp errors for now unless the nsp rules end up pointing
at the nvp.
```

93/02/22
07:51:00

(trngps)(mikey:throg Job: vol.8 Date: Mon Feb 22 07:50:58 1993)

psfilter.in.23447

23

C3800 Troubleshooting Guide

if no error exists on the nsp then failure is either connectivity, a bad nvp, or a bad nsp
try spu4000 -s 812 (to test connectivity)
if no failure is found by spu4000 -s 812 then try swapping nvp with another cpu (if possible), and reboot looking for failure to move, if it does then it's a bad nvp, if it doesn't, then move the nsp between cpus and reboot looking for the failure to move, if it does it's a bad nsp, if not then there is a resistive open/short not detectable by spu4000.

If you only have 1 cpu then I would replace the nsp first and see if the failure goes away, if not replace the nsp and try the nvp. If the failure still persists then there is a resistive open/short not detectable by spu4000.

note: in the case of a resistive open you will need to know what the data held on the nvp was for each failure and check to determine if the same data byte is failing, in which case after board swaps are exhausted you can at least narrow the nets to be checked for connectivity to 9.

ANY OTHER CASE

if any other extractor finds an error besides the ones listed above, then the error is entirely contained on the nvp and the board should be rejected.

13.* SST Discussion

This is a brief overview of sst as it pertains to the C3800. The only executable version of sst is currently v2.2.2. No lower version will execute properly. Additional information can be obtained from the SST Users Manual, part number 760-001530-001.

To install sst it is necessary to perform the following

- 1) cd /sst
- 2) load sst data tape
- 3) installsw -i

To execute sst tests it is necessary to cd /sst and enter "sst" at the spu prompt. This will display the Main Menu as shown below. At the "sst prompt" enter a "4". This will generate a response that asks for the pattern tape to be loaded. This is the same tape that is used to load the sst execution files. The program will then begin to pack patterns of rings that are executable on that particular system. Depending on the number of different part number boards installed, this process will take 20-60 minutes.

NOTE It is possible to pack these pattern and install sst with the system at multiuser.

C3800 Troubleshooting Guide

At the completion of the packing phase, the user will be asked if they wish to execute the tests. If the system is booted it will be necessary to bring it to the spu before proceeding. Execution of the tests will take 6-9 minutes depending on the configuration of the system.

SST Main Menu

- | | |
|--------------|---------------------------------------|
| 1. configure | configuration menu |
| 2. test | test parameter menu |
| 3. display | display parameter menu |
| 4. run | run an sst test for a set of patterns |
| 5. continue | continue an interrupted run |
| 6. retry | retry beginning at first failure |
| 7. save | save the current sst state |
| 8. restore | restore a previous sst state |
| 9. clean | delete all unusable image sets |
| 10. delete | delete an image set |
| 11. do | execute commands from a file |
| 12. help, ? | display help information |
| 13. quit | exit sst |

sst >

Options 1-3 of the main menu are for display of additional menus:

The configuration menu is to be used to alter the hardware configuration to be tested. If a failure is encountered in any ring the only means to execute rings past that point is to exclude the failing ring. This can be accomplished from the configuration menu by entering "-e xx.0" at the sst prompt, where xx is the ring name.

Configuration Menu

- | | |
|-----------------|--|
| 1. display | display the current configuration |
| 2. hardware, -h | set configuration to installed hardware |
| 3. tape, -t | set configuration to rings with networks on tape |
| 4. clear, -c | clear the current configuration |
| 5. add, -a | add scan rings associated with board |
| 6. remove, -r | remove scan rings associated with board |
| 7. include, -i | include rings |
| 8. exclude, -e | do not include rings |
| 9. asm, -m | force board assembly revision |
| 10. wire, -w | force board wire revision |
| 11. part, -t | force board part number |
| 12. main | go to main menu |

The test parameter menu is used to display, or modify test parameters during the execution of sst. Display of submenus will not be listed here as they are self explanatory and would serve only to waste text.

C3800 Troubleshooting Guide

Test Parameter Menu

1. display	display all current test parameters
2. default	default all current test parameters
3. pause	test paus submenu
4. loop	test loop submenu
5. limit	test limit submenu
6. log	test log control submenu
7. main	return to main menu

There are some shortcuts that can be taken when executing sst commands in a very specific troubleshooting effort. For example when executing modules for one specific component, the following command may be helpful:

```
sst conf -c -i ia8 (Where conf is the menu to be used, -c clears the
previous config, -i includes the ia8 for testing.)
This will cause the configuration to be cleared and include only the ia8 on
the next run. This command will leave you in the configuration menu, so to
execute, it will be necessary to quit to the main menu and issue the run
command. Multiple menus can be entered from the same line by separating them
by ";".
```

Adaptations of this method can be very helpful when doing specific testing. When running SST on the entire configuration, this would be of little use.

Below is the coverage provided by sst for the listed boards:

Table 1: SST Stuck-At Fault Coverage

Board	total stuck-at faults	single-board coverage (percentage)	single-board + cross-board coverage (percentage)	cross-board detected faults	scan bits
NVP	1,044,166	74.33%	74.86%	781,662	15,912
NSP	1,029,478	86.69%	87.58%	901,616	18,514
NIA	410,502	76.02%	90.10%	369,862	6,793
NCU	345,318	40.21%	43.46%	150,075	3,039
NMB	290,846	77.41%	79.22%	230,408	9,474
XSl	230,666	51.44%	98.86%	228,036	4,892
XSO	146,972	86.85%	88.34%	129,835	4,166
XRT	99,900	67.23%	85.77%	85,684	4,091
XCL	54,396	18.57%	21.37%	11,624	252

NOTES:

- 1) NMB coverage does not include NMC's. The NMC's are not covered by SST.
- 2) XCL coverage is low due to the fact that it is the scan engine. (theoretically it is covered implicitly by its use in testing the rest of the boards)
- 3) The Single-Board + Cross-Board Coverage Percentage for a given board, is the coverage for that board in the maximum populated configuration for the given machine. So, this number technically only applies to that configuration, but it can also be viewed from the standpoint that this number is possibly considering circuitry that is not being used in a less populated machine.

C3800 Troubleshooting Guide

4) The 4th column numbers are the detected faults. These numbers can be used to generate a system coverage number. Just add up the appropriate numbers from column 1. Generate a sum of the appropriate numbers from column 4, and then divide the 2 summations into each other, and you get your system coverage number.

5) Coverage for a C3880 with 8 memory boards is 80.88%.

6) In general SST uses the longest single scan ring available on a given board. (In the case of the Javelin boards: MCM, MCM3 and PI2, we use the log rings in addition to the main ring, since there is not a scan configuration that lumps that ring into the main ring.)

A.* Version 2.2.3

A new release of SST exists, version 2.2.3, which is intended to extend coverage to the rev H NSP. For all existing systems, the current v2.2.2 will continue to provide adequate coverage. All new systems will ship with rev H NSP's and version 2.2.3 of SST.

It should be remembered that the system has to be in a scannable state to execute successfully. This can be verified by executing a sysreset before execution of sst.

It is recommended, that the set limits off option be used, when executing sst on a C3800. This option is used so that sst will not abort on encountering the first error. As failures are quite often encountered, particularly with the NIA, this will save valuable time in the packing and execution phase.

A log of the SST run is kept in the .log file.

B.* Version 3.0

The newest release of SST (version 3.0) will soon be available in the field.

The primary enhancements in this release are support of APR and decrease of sensitivity to board revision changes.

In support of this there are 2 new entries visible with the cop:

Machine Slot	Device Type	Part Number	Serial Number	Ring Rev	Wire Rev	Assm Rev	DC Rev	AC Rev
sp0	sp	416-006247	1036575	1	F	D	9	0

As seen in the example above, the DC rev alone will now be used to determine the static pattern tests to execute on each board. In the past, it has been far more complex in using all other revision information to make the choice. This will allow hardware changes that don't require pattern changes to be made without affecting SST. This DC rev will be included in the cop info.

The new release will still be downward compatible with previously released boards and will indicate the old method to be used by a DC Rev value of 0. All board releases from now on will require a valid DC entry in the cop chip. A 0 value will not be valid for new releases.

C3800 Troubleshooting Guide

The AC Rev will be used to support CAST based in house testing and will not be utilized in the field.

The second major enhancement will allow testing to be performed online. This is a major component of the APR. This will allow SST to be executed on a board, simply by removing it from the OS. All other heads and memories will remain operational while this is done. A head can be removed from the system by use of the command:

```
cpualloc the -d option will be used to deallocate the head
the -e option will be used to enable the head
```

NOTE It is assumed that the head is in a quiescent state when it is disabled. This is accomplished at the SPU.

The following errors are supported with this release:

- 1 Illegal argument passed to cpualloc
- 2 Unable to reserve specified CPU
- 3 Specified SPU is unavailable
- 4 Init of CPU failed
- 5 sc_init failed
- 6 CPU is not hung, cannot deconfigure
- 7 Cannot configure a running CPU
- 8 Cannot deconfigure a non-running CPU
- 9 Invalid data for head 0 in configuration database.

It should be noted that the xbar, NCU and NIA must be present for operational systems. It will not be possible to remove these boards from the complex, for obvious reasons.

Because of changes to the CDB, it is possible to execute SST on components unassigned to OS simply by entering "run sst" in /sst directory. All files will be packed as necessary and executed on the appropriate boards. Because of modifications, sst cannot crash a system. It is possible to remove a component from OS assignments by the use of xsysconfig.

Booting the system will make all currently assigned components untestable by sst. It will not be possible to remove the component from the complex on line, unless it has failed. If it is desired to execute sst on a component of the complex, that has not failed, it will be necessary to remove it from the complex with xsysconfig prior to booting.

It will, however, be possible the re-enable a downed head by the use of cpualloc and cpuconf.

It is intended that APR will automatically deallocate a failed head and execute sst and if passing will return the head to the complex. The decision to execute SST and the number of failures tolerated are software selectable. Previous SST functionality is supported and can be reviewed in past tech bulletins and the SST Users Guide. Specifically article 21 of chapter 2 of the "C3800 Troubleshooting Hints" will be helpful in this review.

C3800 Troubleshooting Guide

***** NOTES *****

When executing multi board data tests involving the crossbar, failures will occur if hardware in the system has been powered down. This will not occur on single board tests.

When executing multi board tests at the SPU, it is necessary to perform a sysreset before initiating SST testing.

When encountering multiple, between board, failures, it is best to confirm basic continuity of these circuits with spu4000. This is because SST assumes the basic continuity between boards is present.

The following parameter settings should be understood by the user. These are explained further in the "Handbook" and the "SST Users Guide".

- 1) test limit total
- 2) test limit stop
- 3) test limit eliminate
- 4) test limit board
- 5) disp failure pins

14. DDB Primer

Condensed DDB Usage

The ddb utility provides the means to debug the C3800 processor complex, to debug the CPU diagnostics and to access C3800 hardware state via a single utility. To reduce the number of utility command interfaces, ddb has been designed with a user interface which is similar to the command interface of adb. While adb is a software debugger, ddb is a hardware debugger. This subtle difference implies that ddb will not act like adb in all circumstances, nor are all commands in adb supported in ddb.

Scope of this Document

The following information on ddb is a short summary on those commands and operations which have been defined as being required by the Field Service Engineer to effectively trouble shoot a C3800 system. It is NOT an all inclusive User's Guide for ddb, but rather a quick reference. Even the ddb commands covered in the document are not fully explained. Hence for a detailed explanation of DDB, please refer to the internal document "DISE/DDB/DSI User's Guide".

Starting DDB

The ddb utility is invoked by typing "ddb" at the SPU's dsh prompt. When ddb starts up it does a few basic functions. First it determines the first CPU in the complex and assigns that cpuid as the default cpu and the default cir (communication index register), and it sets the default thread id to zero. Next, it sets the default memory accessing mode to use logical addressing. Last it will remove the page_map file if one exists which effectively indicates to ddb, that nothing is loaded into main memory.

93/02/22
07:51:00

(trngps)(mikey:throg Job: vol.8 Date: Mon Feb 22 07:50:58 1993)

26

psfilter.in.23447

C3800 Troubleshooting Guide

On a system which has two heads in ports 2 and 3; which have been successfully powered and initialized, and which are selected for operation from the `xsys_config` utility: the initial `ddb` prompt will look like this:

```
cpu:2,cir:2,tid:0
[DDB]->
```

Please note that `ddb` has a two line prompt. To exit `ddb`, the user need enter the command "`$q`".

Setting DDB Defaults Parameters

Many commands executed under `ddb` operate based on the current set of defined `ddb` parameters. The defined set of `ddb` default parameters are: `default_cpu`, `default_cir`, `default_tid`, and the memory accessing mode.

Changing the default_cpu

The command "`$dcpu #`", where `#` is a valid `cpuid`; will change the default `cpu` for subsequent `ddb` commands. If in our previous example the command "`$dcpu 3`" had been executed, the `ddb` prompt should be changed and displayed as follows:

```
cpu:3,cir:2,tid:0
[DDB]->
```

If the `cpuid` input by the user is invalid (out of range, or the `cpu` is not available for testing), then an error message is displayed and the current `default_cpu` value is retained.

Changing the default_cir

The command "`$dcir #`", where `#` is a valid communication register index; will change the default `cir` for the subsequent `ddb` commands. If in our previous example the command "`$dcir 6`" had been executed, the `ddb` prompt should be changed and displayed as follows:

```
cpu:3,cir:6,tid:0
[DDB]->
```

If the `cir` input by the user is invalid, then an error message is displayed and the current `default_cir` value is retained.

Changing the default_tid

The command "`$dtid #`", where `#` is a valid thread id; will change the default `tid` for the subsequent `ddb` commands. If in our previous example the command "`$dtid 4`" had been executed, the `ddb` prompt should be changed and displayed as follows:

```
cpu:3,cir:6,tid:4
[DDB]->
```

If the `tid` input by the user is out of the 0 to 31 range, then an error message is displayed and the current `default_tid` value is retained.

Changing the default memory accessing mode

C3800 Troubleshooting Guide

When `ddb` starts, it sets the default memory mode to be logical addressing. This implies all C3800 memory addresses are translated using Convex's virtual to physical address translation. The second memory mode is physical addressing. In this memory mode, all memory accesses are treated as physical addresses and no address translation is done. The default memory mode is changed via the command "`$mmode <mode>`", where `<mode>` is either `phys` or `log`. Entering "`$mmode`" with no mode, will cause `ddb` to display to the user the current default memory accessing mode. Below is an example of changing the default memory accessing mode to physical, to logical, and then querying for the current default memory access mode.

```
cpu:3,cir:6,tid:4
[DDB]->$mmode phys
Using PHYSICAL addressing
cpu:3,cir:6,tid:4
[DDB]->$mmode log
Using LOGICAL addressing
cpu:3,cir:6,tid:4
[DDB]->$mmode
Current memory access mode is: Logical
```

Displaying Register Information

The register state of the C3800 complex is divided into 4 categories: scalar information, vector information, communication registers information, and control space information.

Displaying Scalar Information

Two commands have been provide to display scalar information. The first command, "`$r`" will display the scalar register information for the default `cpu`. The second command, "`$R`" will display the scalar register information for each `cpu` which is installed in the system and available for testing. As with our previous example with the default `cpu` set to 3, executing the "`$r`" at the `ddb` prompt will display the scalar register state in the following format:

```
Register state for CPU 3
pc=00000000 upc=00000000 cir=00000000 tid=00000000 ccr=00000000
ps=00000000 (XF)
sp=00000000 a1=00000000 a2=00000000 a3=00000000
a4=00000000 a5=00000000 ap=00000000 fp=00000000
s0=0000000000000000 s1=0000000000000000 s2=0000000000000000 s3=0000000000000000
s4=0000000000000000 s5=0000000000000000 s6=0000000000000000 s7=0000000000000000
t0=00000000 t1=00000000 t2=00000000 t3=00000000
t4=00000000 t5=00000000 t6=00000000 t7=00000000
v1=00000000 vs=00000000 vv=00000000
vm=00000000000000000000000000000000
```

The execution of the "`$R`" command would result in the scalar register state being displayed for all installed and available `cpus` (in our example, `cpus 2` and `3`.)

C3800 Troubleshooting Guide

Displaying Control Space Information

One command is available for displaying the limited about of information which is accessible from the control space. The ddb command "\$xspace" will display this information information is the following predefined format:

```
CONTROL SPACE DATA
SYSTEM INTERRUPTS ARE DISABLED
SYSTEM GLOBAL INTERRUPT ENABLES = 0x00
SYSTEM GLOBAL PENDING INTERRUPTS = 0x00
CPU LOCAL ENABLES:
  CPU0 = 0x00    CPU1 = 0x00    CPU2 = 0x00    CPU3 = 0x00
  CPU4 = 0x00    CPU5 = 0x00    CPU6 = 0x00    CPU7 = 0x00
INTERRUPT BROADCAST ENABLES:
  CHAN0 = 0x00   CHAN1 = 0x00   CHAN2 = 0x00   CHAN3 = 0x00
  CHAN4 = 0x00   CHAN5 = 0x00   CHAN6 = 0x00   CHAN7 = 0x00
CPU CIR:
  CPU0 CIR=0    CPU1 CIR=0    CPU2 CIR=0    CPU3 CIR=0
  CPU4 CIR=0    CPU5 CIR=0    CPU6 CIR=0    CPU7 CIR=0
CPU IDLE STATUS:
  CPU0 NOT IDLE CPU1 NOT IDLE CPU2 NOT IDLE CPU3 NOT IDLE
  CPU4 NOT IDLE CPU5 NOT IDLE CPU6 NOT IDLE CPU7 NOT IDLE
```

Display Cache Information

The C3800 machine has a data cache, a pte cache, an instruction cache, and a return control queue which may need inspection as part of normal machine debug. Because the of the detailed information contained in each of the caches, only the commands to display the data is described here.

Displaying the Data Cache

Two ddb commands are provided for display data cache information: "\$dcache" and "\$Dcache". The "\$dcache" command displays the entire contents of the data cache on the cpu identified by the default cpu. The "\$Dcache" command displays the data cache for each cpu installed in the system.

Displaying the Instruction Cache

Two ddb commands are provided for display data cache information: "\$icache" and "\$Icache". The "\$icache" command displays the entire contents of the instruction cache on the cpu identified by the default cpu. The "\$Icache" command displays the instruction cache for each cpu installed in the system.

Displaying the PTE Cache

Two ddb commands are provided for display data cache information: "\$ptecache" and "\$Ptecache". The "\$ptecache" command displays the entire contents of the pte cache on the cpu identified by the default cpu. The "\$Ptecache" command displays the pte cache for each cpu installed in the system.

C3800 Troubleshooting Guide

Displaying the Return Control Queue

Two ddb commands are provided for display data cache information: "\$rcque" and "\$RCque". The "\$rcque" command displays the entire contents of the return control queue on the cpu identified by the default cpu. The "\$RCque" command displays the the return control queue data for each cpu installed in the system.

Displaying C3800 Main Memory

The ddb command to display memory takes on the format "address/format". The address provided by the user is either a physical or logical address based on the memory accessing mode. The "/" is a required specifier which identifies to ddb that the access is a memory access. The size specifier indicates the size of the access and how many bytes to display. Currently, b, h, w, and l are valid.

Loading cputests

For a user to load a cpu test into Convex main memory, the command "\$stest" is provided. the actual syntax for this command is "\$stest testname segment" where testname is cpu4030, cpu4041, cpu4241, cpu4331, cpu4332, and cpu4333, and segment is a value 0 through 7. Execution of the command performs the operations of loading all modules for the given cptest, correctly linking them together, initializing all cpu global variables, and initialization of the communication registers.

A second ddb command "\$clear" is provided to remove a previously loaded cptest from the internal ddb structures. For example if "\$stest cpu4030 0" is executed, then ddb know about cpu4030. If another "\$stest cpu4030 0" is executed, a collision will occur for the logical addresses. Hence if a "\$clear" is executed before the second "\$stest", then all knowledge of the previously loaded test is removed.

Another ddb command which is helpful is the "\$dmem" command. This command will dump how the cptest was loaded into main memory, what logical addresses and physical addresses the code segment were loaded into and physical addresses of the pte tables.

Displaying CPU status

The command "\$status" is provided to give the user a complete view of how ddb and the CPU environment are setup. When the "\$status" is executed, a display similar to the following is presented to the user:

C3800 Troubleshooting Guide

MEMORY MODE	STOP MODE	BKPT MODE
LOGICAL	NORMAL	DIAG
CPUID	CPUSTATE	CPUMODE
CPU0	UNINST	PARKED
CPU1	UNINST	PARKED
CPU2	HALTED	IDLE
CPU3	HALTED	IDLE
CPU4	UNINST	PARKED
CPU5	UNINST	PARKED
CPU6	UNINST	PARKED
CPU7	UNINST	PARKED

```
Test name = 'NO TEST SPECIFIED' is loaded into segment '0'
data_ffaults = 'OFF'          ip_ffaults = 'OFF'
enable_dcachc = 'ON'          vl_count = '16'
seq_mode = 'OFF'              chain_mode = 'OFF'
parallel_mode = 'OFF'         multi_cirs = 'ON'
scn_ovr = 'ON'                secure_mode = 'OFF'
```

In reading this status display, the user can see the memory access mode is logical, CPU's 0, 1, and 4 through 7 are uninstalled (hence they considered parked), CPU's 2 and 3 are installed with clocks HALTED, and no cputest has been loaded. In the fields below "Test name" are the CPU test parameters. The values of each the fields can be altered by preceding the parameter name by a \$ and followed by a value. For all parameters except vl_count, the value is 0 or 1. For vl_count, the value is 0x0 to 0x80.

Configuring CPUs for testing

As shown in the \$status display, there are fields for each possible CPU in the complex. The first field is the CPUSTATE: this field indicates if a cpu is installed in the system or not. If a cpu is installed, then the actual state of cpu clock are displayed (HALTED or CLKS_ON.) Hence, the user should always see one value UINST, HALTED, or CLKS_ON for the CPUSTATE on each CPU. The field to the right of the individual CPUSTATE is the CPUMODE. If a head is uninstalled, this value will always be PARKED. If a head is installed, the head is under the control of the user.

For a user to run a cputest from ddb, the first decision is to determine which CPU's are to be executed. In our example, if the user wants to only test CPU2 and not CPU3, then executing the commands "\$run 2" and "\$park 3" will park CPU3 and place CPU2 in the run mode. Now when a subtest is started, clocks will only be issued to CPU2. For cputests cpu4030, cpu4041, cpu4241, cpu4331, and cpu4332, the user should always set the heads to be either PARKED or RUN (never IDLE.) For cputests cpu4333, the user should select one cpu to be in the RUN mode and then select all other CPUs to be in the IDLE mode. In our example, if we wanted to run CPU3 as the controlling CPU, we would execute the ddb commands "\$run 3" and \$idle 2" to place CPU3 in the RUN mode and CPU2 in the IDLE mode.

C3800 Troubleshooting Guide

Parking a CPU

The ddb command "\$park #" is used to place an installed CPU into a mode where it will not be tested. This result of executing this command will cause the identified CPU to be placed in the PARKED MODE when the CPUTEST is started.

Idling a CPU

The ddb command "\$idle #" is used to place CPU# into the idle loop when a CPUTEST is started.

Running a CPU

The ddb command "\$run #" is used to place CPU# in the run mode when a CPUTEST is started.

CPUTEST Execution

Executing a subtest of a loaded CPUTEST is a matter of entering the ddb command "address:r" where address is the address where the test is to start. This address should be consistent with the memory accessing mode. In addition, if the address is a logical address, then a symbolic name can be entered for the address. When the user enters "address:r" ddb is going to block user input during the execution of the test. This allows input to ddb to be redirected from a file. If the user wants ddb to not block during the subtest execution, entering a ^C will unblock ddb.

CPUTEST Termination

Any subtest in a CPUTEST will exit in one of 3 ways: it can pass (indicated by a 0x100 in a1), it can fail (indicated by anything other than 0x100 in a1), or it can cause a harderror. If a subtest passes or fails, the control of ddb is returned to the user, the contents of a1 are displayed, and the instruction currently pointed to by the program counter is displayed. If a harderror is generated, then the hard_logger is executed to dump the cause of the hard errors and control is returned to the user. The other condition which is user can encounter with ddb is a subtest hanging. When this condition occurs, the user should enter ^C to unblock ddb command input and then enter the "\$status" command to determine the state of the CPU's. If a CPU is found to have its clocks on (CLKS_ON), the executing the "\$halt #" will turn off clocks for CPU#. This is necessary before any register state on the CPU can be inspected.

CPUTEST Example

The following is a set of examples which the TAC had previously supplied to the field to convey how to run cputests via ddb. This information is being included to further clarify the operation of ddb for the field.

1. Run cpu4041 in Segment 0, data forced faults, ip forced faults, vl of 0x10, and chain mode on heads 0 and 1 the user would enter the following ddb commands:

C3800 Troubleshooting Guide

```
cpu:6,cir: 6,tid: 0
[DDB]-> $q
```

6. The following is an example of a cpu4041 test in the chained mode.

```
spu> ddb
cpu:6,cir: 6,tid: 0
[DDB]-> $test cpu4041 0
loading p0r0_4041.ldr at logical offset 0x00000000
loading cpu4041.ldr at logical offset 0x00008000
Test loading completed, initializing Comm Regs
Comm Reg initialization completed, setting up SDRs
cpu:6,cir: 6,tid: 0
[DDB]-> $run 6
$run: CPU6 placed into run mode
cpu:6,cir: 6,tid: 0
[DDB]-> $spark 7
$spark: CPU7 placed into parked mode
cpu:6,cir: 6,tid: 0
[DDB]-> $chain_mode 1
chain_mode = 'ON'
cpu:6,cir: 6,tid: 0
[DDB]-> Chain_entrypoint:r
CPU6 halt: a1=00000100 PASSED
Chain_entrypoint_end: halt #0x100,a1
cpu:6,cir: 6,tid: 0
[DDB]-> $q
* * * * * Last command returned status 1 * * * * *
spu>
```

15.* Scan Based Test Failures With REV H NSP's

Now that the rev H NSP's are arriving in the field as replacements, it should be understood that certain failures can be obtained with scan based tests such as spu4000 and SST.

The diagnostic spu4000 will fail subtest 710 and subtest 810 if run with all available boards installed. If spu4000 is executed on only the rev H NSP and necessary other components (i.e. ncu and xbar) then it will pass.

Version 2.0 of diagnostics and 2.2.3 of SST will rectify the problem. Although, even with 2.0 Diagnostics installed, it will still be possible to experience failures with subtest 710.

It may not be possible to run spu4000 successfully when multiple rev H NSP's are installed in a configuration with other lower rev NSP's and running v1.1.2 diagnostics.

The recommended procedure for executing spu4000 when rev H NSP's are installed, is to configure all boards except NVP's. Leave the NVP's out of the configuration. After the test is run, then disable NSP's and memory and enable the NVP's and execute spu4000 subtest 810. This will afford the best chance of success.

C3800 Troubleshooting Guide

It is still possible to encounter failures with subtest 710, 713 and 810. Although, with a single rev H NSP, this is unusual.

In the case of multiple rev H NSP's, it may require powering the rev H boards down and executing the tests and then powering the other heads down while executing on the rev H boards.

If problems are encountered executing spu4000, it will be necessary to completely exit the diagnostic before attempting to execute again.

The important thing to remember is that if failures are encountered with spu4000, it may be due to rev H NSP's in the system.

Because of these failures, an attempt will be made to expedite v2.0 of the diagnostics and 2.0 of spu unix.

16.* C3800 Diagnostics Version 3.0.0.6

Now that the revision 3.0.0.6 Diagnostics is being released, it's time to explain some of the differences and enhancements produced for this release.

This revision will be required for support of APR and the version 3.0 SST which will be explained in other articles.

Enhancements:

- 1) add_parts - A new utility that interrogates the /diag/db/part_numbers file and makes necessary additions to the Configuration Data Base.
- 2) cop/copmod - Modification used to support changes in 3.0 of SST. The new cop and copmod will now list DC and AC rev levels for all boards. This info will be used to determine the correct SST patterns to run on boards, rather than the old method utilizing ring asm and wire rev's to determine board pattern type.
- 3) ddb - Added -i option that allows ddb to use current page_map info to allow ddb to be used on currently loaded CPU jobs.
Added breakpoint capability within CPU code.
Supports trace traps.
- 4) Diaginit - Checks installed NSP's for support of APR. Will update the dynamic_cpu_harderrs_n entry in the CDB.
- 5) Errintd - Changes required for APR.
 - a) Includes, only, non APR functional heads in hard error mask.
 - b) Added support to check for hard errors on heads that are APR functional.

C3800 Troubleshooting Guide

- c) Places APR related messages in the errlog.
- d) Added support for XCL2 to clear soft errors after processing the error. This will allow additional soft error support on the XCL, but is not required for APR.

6) fw_rev_update - Updated to reflect new loaded firmware which is 3.16. The new bpc_firmware fixes a problem of processing non existent bays and timeout failures between ppcs and bpcs.

ppc firmware is now at 2.8 and fixes problem of report of 300v out of range continuously. Now reports once.

- 7) hard_logger - Adds -m option to process CPU's that do not have APR function enabled.
- 8) mmnit - Modified to detect 1/4 populated NMC's
- Added -t option which denotes timing info in CDB.
- All changes to support NMB2.

9) CDB entries for APR support:

enable_cpu_harderrs
cpu_hard_err_sniff_period Discussed in APR Article
dynamic_cpu_harderrs_n
cpu_automatic_realloc_n

Other new entries:

- mb_installed_dram_sizes_p_n (n=0-7) Specifies size of drams on NMC.
- mb_installed_dram_rows_p_n (n=0-7) Specifies number of rows on dram on NMC.
- head_0_mm Data captured from SP0 after a hard err is detected.
- bank_intrlv_sel_n (n=0-4) Specifies interleave where 0=1 bank 1=2banks, 2=4 banks, 3=8 banks and 4=16 banks
- cpu_running_os_n (n=0-7) Denotes cpu's running OS.
- enable_1x_purges Enables a 1x purge of rams instead of 2x.
- 10) ucode_rev_update - SR is now at 10.6 and US advances to 10.36. Others remain at 10.0. Changes support APR.
- 11) xsys_config - Added capability to disable cpu hard errors. A new submenu that allows users to specify bank interleaving.
- 12) mem4000 - Support NMB2
- 13) spu4000 - Updates substest 810 to test CPU backplane hardware change required for APR.

C3800 Troubleshooting Guide

Fixes:

- 1) bpcwachd - Modified error handling to match firmware.
 - 2) copmod - Allows CCU's to be cop modified.
 - 3) cpualloc - Fixed to allow heads to be dynamically added and removed from the system with OS running.
 - 4) ddb - Fixed interaction problem with VM register.
 - 5) diaginit - Modified to set osc_freq in CDB to 16.7ns after oscillator speeds have been determined.
 - 6) errintd - Modified so errors processing memory soft errors do not cause termination of errintd.
 - 7) hard_logger - Screen saver option starts after keystroke or mouse event rather than waiting for next boot.
 - 8) swix_load - Corrects SWIP download failures.
 - 9) mmnit - Logs errors in event logger. Initializes new memory related entries in CDB. Will indicate actual size of NMB.
 - 10) mem4000 - Fixes substest 366 to allow different size NMB's to be tested.
- Fixes substest 500 to test all of memory.
- Fixed error checking failures in substest 366.
- 11) xdiag/cpuucti - Fully functional and support all CPU diagnostics.

Utilities fixed and/or modified in this release:

- 1) copmod
- 2) cpualloc
- 3) errintd
- 4) mmnit
- 5) xsys_config
- 6) cop
- 7) cop contents

93/02/22
07:51:00

(trngps)(mikey:throg Job: vol.8 Date: Mon Feb 22 07:50:58 1993)

psfilter.in.23447

33

C3800 Troubleshooting Guide

17.* spu4000 Faults

The following are C3800 nets recently reported by manufacturing as not being detectable by spu4000.

1. spu4000 passed but sys_con detected the following:
ia8_xsle.wr_data[31..0] to reg xsle:wr_data_is8 = 0x200000 (expect 0x0)
2. spu4000 missed these shorts that xc_con found;
ERROR: XC_IA8.TRAP_VECT<7..0> to register ia8:xbi_ncu_intvec_7_0 = 0x57 (expect 0x55)

ERROR: XC_IA8.TRAP_VECT<7..0> to register ia8:xbi_ncu_intvec_7_0 = 0xae (expect 0xaa)
3. Spu4000 did not detect the following shorts that were reported during ConvexOS boot.
VP6.3_414 (SPOVPODATA35) VP6.3_314 (VPOSPDATA35)

18.* spu4000 Bug

If the scalar boards in a C3800 system are revH or later, there's a known problem with the diagnostic when all installed processors are running at 810 at the same time. It shows up as a vp-sp hard error failure.

If the heads are run separately the error will not show up, if that's what the problem is.

Here's an example of the known failure:
Running Vector Processor 0 to XCL Test.

```
+++>
<Thu Jul 30 14:52:30 1992> /diag/test/spu/spu4000:../vp_xcl_test.c:273
Subtest Fail (DiagER466): Spu4000 Error: Connectivity Test Failure
```

```
Detected Short or Stuck-at-1
Source Signal: VP_SP.HARD_ERROR          Source Port: 0
Sink Signal:   VP_SP.HARD_ERROR          Sink Port:   1
Sink Field:    xcls:CU_TRAP_COMP_SPX
Falling Bit Position: 0
Expected:      00000000
Actual:        00000001
****
```

19.* Rev H NSP failures With 2.0 Diagnostics

It has been discovered that newly upgraded NSP's will not function in a field system, running on 2.0 Diags.

The reason for this is related to two (2) extra fields generated by the 3.0 version of copmod. These two extra 32 bit fields are not recognized by the 2.0 diagnostics. All boards in house are now copped with 3.0, so this affects all boards, now being shipped.

C3800 Troubleshooting Guide

The solution to the problem is to, immediately, upgrade all C3800's to 3.0 diagnostics and 10.2 OS. In addition, at the same time, it is recommended that the backplane upgrade (FMI 93) be performed at this time. This is recommended in order to avoid any other hidden problems.

It is important that both 3.0 and 10.2 be loaded together, as loading 3.0 alone can result in hangs, due to APR problems. In an emergency, this interdependency can be circumvented by enabling the xsys_config entry "enable_cpu_harderrs". This entry is located at the bottom of the xsys_config display. If on then APR is defeated, if off then APR is enabled. It is controlled by the mouse.

It should be understood that it will not be possible to install any replacement boards, until this is accomplished.

For more details on APR, please refer to the tech bulletin, issued on that subject.

It is urgent that this be accomplished, immediately. Please, don't get caught in a situation where this is necessary after the system fails.

20.* Diagnostics version 3.1

The 3800 diagnostic release 3.1 will soon be released. The following is a list of enhancements and changes for this release.

Please, understand that ppc and bpc firmware is contained in this release, so allowances should be made for time to install. The firmware for each module will take approximately 3 1/2 minutes to download.

SST version 3.1 is a prerequisite for this release. Please ensure that SST is loaded first, or remove the old version before loading 3.1 Diagnostics.

POWER SYSTEM CHANGES:

1. Firmware

BPC firmware is now at revision 3.17 - changes the default bay inlet and outlet temperatures:

Inlet temps - warm=30 deg C and hot=35 deg C

Outlet temps - warm=45 deg C and hot=50 deg C

PPC firmware is now at revision 2.10: fixes primary power (300v) out-of-range problem. Caused message queue to fill up and hang the power system.

2. Utilities

altsetpts allows user to specify the bay inlet warm/hot temperatures and calculates the outlet warm/hot temperatures (+15 deg C). For example,

93/02/22
07:51:00

(trngps)(mikey:throg Job: vol.8 _ate: Mon Feb 22 07:50:58 1993)

psfilter.in.23447

34

C3800 Troubleshooting Guide

altsetpts -t -w 32 -h 38 bay4
sets the inlet warm temp to 32 deg C and the inlet hot temp to 38 deg C.

diaginit no longer displays the full bay status. Use -v option to get the full report.

/diag/db/set_busses now sets the vtt on the MB board to -2.00V (was -1.95V).

bpcwatchd has been modified to correctly handle "power off" unsolicited messages from the ppc. Now cleans up the Configuration Database to indicate the ppc is powered off.

Software support for the new keyswitch.

UTILITY/TEST ENHANCEMENTS:

1. Utilities

Added part numbers for RTIOP, newm idc/itc board and rev C xbar backplane.

Modified the SPU kernel to fix the window allocation problem (these are main memory windows).

Added timestamps to cpualloc output for aid in determining down time.

IA and SP soft errors are now stored in the Configuration Database and can be displayed using dump_soft_log.

hard_logger now invokes the system_info script when processing APR errors. This includes the same data as when processing a "normal" hard error.

Added call to the cleanup command in the initall script.

Added the /etc/shells file to allow users to ftp into the SPU.

2. Diagnostics

io5000 now supports the RTIOP board.

Added additional short checking to the sp<->vp and ia8<->xbar tests in substest 810 of spu4000.

Added source/sink signal bits to spu4000 substest 810 error displays.

Added checking so cpu4333 can be run on a single head - cpuct1 deselects substests that require 2 or more heads.

UTILITY/TEST FIXES:

1. Utilities

cpualloc does not clear the cpu_os_req_? flag - allows head to come on-line on next reboot of ConvexOS.

C3800 Troubleshooting Guide

Errintd

No longer attempts to log soft errors from a disabled head

Now disables APR when only a single head exists. APR is re-enabled when another head becomes available.

Now correctly invokes the hard_logger when more than 2 heads pull a hard error.

Fixed initialization of 8247 type SPs in sysreset to enable soft errors - no longer need to edit the /diag/db/scn_ovr file.

Fixed the part number entry for the VIOP board - now accepts all assembly revs.

Fixed ddb so displays tregs in the register display.

Mminit

Fixed intermittent sizing problem - will not need a 2nd run.

Disallows downsizing of 256MB size boards to 128MB.

2. Tests

Fixed timeout problems in cpucti when running cpu4331 in chain mode.

The SPU version of idcfmt now works, i.e. can format a disk.

Other fixes:

cpualloc.test now returns a 0 if no SST data patterns exists. Head will come on-line even if SST is never run.

The correct version of scan_trace is now installed - this is the 1X version.

Updated version of nts.

Can now specify parameters to mtst on command line:

mtst -d <addr> <byte_cnt> <xfr_type> <opcode> <even_pat> <odd_pat>

adb is now part of the diagnostic release.

CXTS SUPPORT: (CXTS will not be available in the field before December)

All error logging now uses RPC instead of system V IPC.

xsfp has a new boot option: cxts control boot.

xsfp starts 3 new daemons:

cxts_rt.prt

93/02/22
07:51:00

(trngps)(mikey:throg Job: vol.8 Date: Mon Feb 22 07:50:58 1993)

psfilter.in.23447

35

C3800 Troubleshooting Guide

cxts_server.prt

cxts_ui

If CXTS daemons are not found, i.e. CXTS is not loaded, an error is printed in SPU Console window.

If CXTS daemons are found then startup of X environment after a SPU boot will take longer.

The "RPC: Program not registered" message during SPU boot can be ignored.

Any compiled user programs that log events need to be recompiled.

21.* SST Version 3.1

Version 3.1 of SST will soon be released. The primary reason for this release is for support of CXTS. In fact this is a prerequisite for 3.1 Diagnostics.

This is because the libraries are being converted from IPC (Internal Process Control) to RPC (Remote Process Control). If 3.0 SST is executed with 3.1 diagnostics then a hang will occur. If 3.1 SST is executed on 3.0 Diagnostics, the error messages "RPC:Program not registered" will occur during execution.

22.* spu4000 and Intermittent Failures

It should be understood that spu4000 should not be relied on when trying to solve intermittent failures. This is because spu4000 is not an at-speed test and therefore will not detect subtle point-to-point net failures.

It takes a 21-25 ohm net to actually create a large enough voltage drop to be detected by spu4000. Any net above 2.6 ohms will present itself as a potential failure in an at-speed condition (i.e. booted). Therefore, eliminating a connectivity failure as a possible source of an intermittent problem, because spu4000 finds nothing, can be a serious mistake.

A few ohm increase in point-to-point resistance will still not be sufficient to cause the termination resistance to vary outside of normal parameters, of 50-56 ohms. Therefore the only reliable measurement on intermittent nets is point-to-point.

C3800 Troubleshooting Guide

CHAPTER III
(C3800)

1.* C3 Bay Configuration Considerations

Although there is no functional reason that the C3 bays cannot be configured in any manner, or combination necessary, there are several considerations that should be addressed prior to using a nonstandard bay arrangement:

- 1) It would be difficult to determine the exact physical configuration when dialing in by personnel not familiar with the site. This could lead to less than optimum troubleshooting decisions.
- 2) Although all ports will be tested, the most thorough testing will generally occur while configured in the standard manner.
- 3) If the configuration is done purely for appearances, then when upgrades are performed, it may be necessary to reconfigure the entire system again. This can result in introducing additional failures and increasing the work load.
- 4) If the system is installed in a T configuration, then removing the power harmonizer from either CPU bay will result in one of the bays having to be totally removed from the configuration for harmonizer removal.

By considering the situation from this perspective will lead to a more satisfied customer and Field Engineer.

2.* C3 I/O Bulkhead Boards

It has been decided that CONVEX will begin shipping I/O bulkhead boards in all slots on all C3 shipments.

In the past only the bulkheads for the required configuration were being shipped. This meant that the cables were run from the CCU's and tied up where they would attach to the bulkheads. This could cause some difficulty when it was necessary to upgrade the I/O on the 3800.

From this point forward, a minimum of one additional IDC Bulkhead will be shipped. This will always be installed on the Pbus 2 and 3 side to allow some troubleshooting leverage.

It is hoped that this will make future upgrades easier for the field.

C3800 Troubleshooting Guide

3.* C3 Purge Ram Function

There have been numerous questions regarding purge rams and this will be an attempt to shed some light on the subject.

The purge rams are a mechanism of the Neptune cache system that is used to track the use and validity of the individual cache data. These purge rams are very similar in function to the Tag and Validity Rams that are used in the C2 RITA circuitry. The major difference between the C2 and C3 are that in the C2 individual invalidation and replacement was used in the cache, but in the C3 the entire cache is purged. The purge operation will be performed in one clock cycle.

All purge rams are located on the NSP and there are 19 of these devices located on each NSP.

These purge rams can be identified as any net or signal that contains a reference to tag, validity, update, or history. For example: the error npal.pval_par_err is an example of a purge ram parity error. Note the second entry pval contains the reference to validity.

4.* C3800 Gate Array Functions

The following are a list of gate arrays used in the C3800. Included will be a short functional description and in parentheses will be the number installed on that board.

Located on NVP

- NVD (1) Vector Dispatch -- Vector Dispatch logic receives instructions from the IP on the NSP.
- NIS (2) Input Staging -- The Input staging gate arrays handle data from the NSP and memory. Referred to as IS0 and IS1.
- NVRF (8) Vector Register File -- Eight 128X64 vector registers labeled V0-V7. Same as C2.
- NVM (3) Pipe Controllers -- Controls Add, Multiply and Load pipes.
- NQ (3) Pipe Control Queues -- One for each pipe.

Located on NSP

In NAS Section (Address and Scalar Data Path)

- NUS (1) Micro Sequencer -- Controls AS. Converts instructions from IP to microinstruction stream for execution.
- NRFA (4) Register File -- Main Integer execution unit. Contains
 - 8x32 Address registers (A0-A7)
 - 8x64 Scalar Registers (S0-S7)
 - 8x32 Temporary Registers (T0-T7)
 - Hazard Logic

C3800 Troubleshooting Guide

- NPSW (1) Program Status Word -- Controls flow through functional Units.
 - NMISC (1) Miscellaneous Functions -- Floating point to/from integer conversions. Leading ones, trailing zero, shift and population count.
 - NFAD (1) Floating Point
 - NMUL (1) Floating Point Multiply
 - NDIV (1) Division and Square Root
- In NIP (Instruction Processor)
- NIAD (2) Instruction Processor Address Generation -- Address
 - NPAR (1) Instruction Parser -- Cracks Entry Points, register fields, and other info from the Icache.
- In NDC (Data Cache)
- NDC (1) Data Cache Control
 - NDP (3) Data Path -- Stages data for the DC
 - NAG (2) Logical Address Generation -- Stages Addresses for the DC
 - NPA (2) Physical Address Generation -- One even and one odd.
- In NRC (Return Control)
- NRC (3) Return Control

5.* New hard dumpers for known failures

The new dumpers will be executed by the hard_logger when the particular crash is encountered. The output of these dumpers will not be saved in the errlog but will appear in output files in /sst as follows:

- 1) for purge ram errors 2 files are generated. pram.log.'date' & pram.stdout.'date' where date is the actual date.
- 2) For nrvf failures the file is nrvf.out.'date'.
- 3) For dcache stam failures the file is dcache.out.'date'
- 4) For icache stam failures the file is icache.out.'date'
- 5) For ptecache stam failures the file is ptecache.out.'date'

After a crash involving one of the above, it will be necessary to forward the contents of the appropriate file and the errlog to the TAC as soon as possible.

C3800 Troubleshooting Guide

6.* C3800 Firmware Update Procedures:

In order to install and execute new firmware for the power subsystem, it is necessary to install the new firmware and make the master changes in the file "/diag/db/fw_rev_update, which appears as follows:

```

'/diag/bin/cdb_update bpc_master_fw_rev "2.14"
'/diag/bin/cdb_update enforce_fw_revs 0';;
'/diag/bin/cdb_update bpc_master_fw_rev "3.15"
'/diag/bin/cdb_update enforce_fw_revs 1';;
'/diag/bin/cdb_update ppc_master_fw_rev "2.7"
    
```

After making the ppc or bpc master entry necessary then the script should be executed. This will update the cdb file.

To upgrade the ucode the process is similar. Make the appropriate entry in the file "/diag/db/ucode_rev_update" and execute it. This file appears as follows:

```

/diag/bin/cdb_update master_sr_rev 1.15
/diag/bin/cdb_update master_us_rev 4.78
/diag/bin/cdb_update master_ua_rev 1.2
/diag/bin/cdb_update master_ul_rev 1.1
/diag/bin/cdb_update master_vd_rev 1.2
    
```

It should be understood that changes made directly to the cdb file, using cdb_update, or cdb_browser will not be permanent.

7.* Displaying The C3800 System Serial Number

To display the system serial number on a C3800 it is only necessary to perform the following:

```
get mach_serial_number
```

This will then return with a hexadecimal display of the serial number. The value will appear as a0xx.

8.* C3800 osc_freq and Released NCU's

With the release of 3.0.0.6 diagnostics, the osc_update utility in /diag/db has been deleted. This is due to an effort to fix the nominal clock output at boot. This will obsolete the article 8 of chapter 3 in the C3800 Troubleshooting Guide.

In order to change the clock frequency under 3.0 diags, it is necessary to accomplish:

C3800 Troubleshooting Guide

put osc_freq x <where x is a value from 0-2>

- 0 = nominal 16.7ns
- 1 = lower 18.0ns
- 2 = upper 15.5ns

To display the current clock value a get of osc_freq will produce the following display:

```

VALUE: 16#00000000
|31|30|29 | 28 # 27 | 26 | 25 | 24 # 23 | 22 | 21 | 20 # 19 | 18 | 17 | 16
|                                     Reserved                                     | spare
|
|15|14|13 | 12 # 11 | 10 | 9 | 8 # 7 | 6 | 5 | 4 # 3 | 2 | 1 | 0
|                                     Reserved                                     | osc
    
```

It is important to remember that a diaginit, or a reboot of the Opus will return the value of this register to 0.

9.* C3800 STRAMS

The STRAMS used on the C3800 refer to (Self Timed Rams). They are called this because the components require no external clocks, as standard DRAMs do. This is beneficial because it allows these components to be fully scannable.

The STRAM failures have become the predominant problem with the C3800. The reason that they have become the prevailing failure is due to the other problems being brought under control.

The STRAM's are generally used in the caches and control stores on the NSP. They have their own power bus, 10KVEE (-5.2VDC). STRAM's are also used on the NCU in the comm registers and on the NIA in the data buffers.

Below is a list of STRAM associated hard errors. Due to the nature of the failure, a single occurrence of a STRAM hard error is usually sufficient to justify replacing the NSP.

```

STRAM WRITE ERRORS (errors detected by strams - input data)
sp_dcd0_wr_par_err          -dcache data
sp_dcd1_wr_par_err          -dcache data
sp_dct0_wr_par_err          -dcache tag
sp_dct1_wr_par_err          -dcache tag
sp_dcu0_wr_par_err          -dcache update
sp_dcu1_wr_par_err          -dcache update
sp_icd_wr_par_err           -icache data
sp_ict_wr_par_err           -icache tag
sp_lat_wr_par_err           -icache lookaside tag
sp_pted0_wr_par_err         -pte cache data
sp_pted1_wr_par_err         -pte cache data
sp_ptet0_wr_par_err         -pte cache tag
sp_ptet1_wr_par_err         -pte cache tag
sp_srd_wr_par_err           -sram data
    
```

93/02/22
07:51:00

(trngps)(mikey:throg Job: vol.8 _ate: Mon Feb 22 07:50:58 1993)

psfilter.in.23447

38

C3800 Troubleshooting Guide

```
STRAM READ ERRORS (errors detected by device receiving stram data- output data)
sp_nrfa.extdata_par_err      -sram data
sp_npa0.ctag_par_err        -dcache tag
sp_npal.ctag_par_err
sp_ndp0.cupd_par_err        -dcache update
sp_ndp1.cupd_par_err
sp_ndp.yuvc_crd_err        -dcache data
sp_nrfa0.nrfa_wcs_par_err    -wcs
sp_nrfal.nrfa_wcs_par_err
sp_nrfa2.nrfa_wcs_par_err
sp_nrfa3.nrfa_wcs_par_err
sp_npsw.npsw_wcs_par_err
sp_nus.nus_wcs_par_err
sp_nag0.uirl_par_err
sp_nagl.uirl_par_err
sp_ndc.uirl_par_err
sp_npa0.pdata_par_err      -pte cache data
sp_npal.pdata_par_err
sp_npa0.ptag_par_err        -pte cache tag
sp_npal.ptag_par_err
sp_npar.inst_tag_par_err    -icache tag
sp_npar.look_tag_par_err    -icache lookaside tag
sp_npar.que_par_err         -icache data
```

10.* C3800 pram.log Created by New hard_logger Dumpers

Listed below is the contents of the pram.log created by the new hard_logger dumpers on the C3800. This includes the error output from all 19 purge rams. This list includes their names and an error.

As can be seen the error resides in the purge ram ic_valid at location 7b0. The data (1) the data and address validity differ. This should not occur, so the only valid values in these 2 bit registers is a 0, or a 3, indicating the data and address are either valid or invalid. The error occurs when the address and data entries disagree with one another.

The error can be verified by checking the location indicated in the affected ram. A collection of all ram outputs is created in the file pram.stdout. To locate the failure, simply search on the ram name and then the address in the file pram.stdout.

Included with the pram.log is a comment for each ram indicating actual function. These definitions are not included in the actual dump.

```
Ram : la_valid      Look Ahead Validity
Ram : ic_valid      Instruction Cache Validity
      7b0  1
Ram : br_hist_l     Branch History Cache read data lower word
Ram : br_hist_xl    Branch History Cache read data lower word
Ram : br_hist_u     Branch History Cache read data upper word
Ram : br_hist_xu    Branch History Cache read data upper word
Ram : pte_modl      PTE Cache odd modified bits
```

C3800 Troubleshooting Guide

```
Ram : pte_refl      PTE Cache odd referenced bits
Ram : pte_tvall     PTE cache odd thread validity
Ram : pte_vall      PTE cache odd validity
Ram : pte_mod0      PTE Cache even modified
Ram : pte_ref0      PTE Cache even referenced
Ram : pte_tval0     PTE Cache even thread validity
Ram : pte_val0      PTE Cache even validity
Ram : dc_tvall      Data Cache odd thread validity
Ram : dc_vall       Data Cache odd validity
Ram : dc_tval0     Data Cache even thread validity
Ram : dc_val0       Data Cache even validity
Ram : sr_val        Scratch Ram validity
```

11.* C3800 Front Panel Switch Problem

It should be understood that the current front panel switch key is only removable in the off position and not in the secure position. This is an oversight and is being corrected with the release of a new switch. This switch will be cut in on an ecn and will be installed in all currently installed systems.

12.* C3800 diaginit abort

If for any reason that the diaginit process on the C3800 should abort prior to initializing the CCU's, they may not be recognized on the next pass of diaginit. This means that the system will not boot because the CCU's are offline.

This happens because the CCU's share power with the NIA and this has been initialized on the previous pass. In order to get the CCU's initialized on the second pass of diaginit, it is only necessary to powerdown the NIA and then execute diaginit. This will pickup the missing CCU's.

With the installation of v1.0 of the diagnostics, it is a very common situation for the diaginit process to abort after the spu4000 subtests are executed. This is not a cause for alarm and only requires diaginit to be rerun. But in these situations the above information will keep you from having to accomplish a sys_shutdown.

Although fixed in v1.1 of diagnostics, it is still possible to occur and so should be kept in mind.

13.* C3800 Gate Array Identification

In cases where there are multiple gate arrays, or rams for even and odd data, the gate arrays are distinguished by a 0 and 1. For example the NPA0 and NPA1. The convention is that the 0 indication is for even and the 1 is for odd.

C3800 Troubleshooting Guide

The lone exception to this rule is the NDAT, located on the NCU. This designation is reversed for this device. So, it should be realized that NDAT1 handles even data.

14.* NCU Powered off and mminit Failures

It should be noted that if the NCU is powered off and/or removed from the system individually, there is a possibility of encountering mminit failures after the diaginit.

The mminit failures will be due to a failure with access of the memory test logic (MTL) located on the NCU. This failure can be overcome by executing a sys_shutdown and re-executing diaginit.

15.* Memory Initialization Failures

It should be understood that the mminit process, on the C3800, is accomplished by the Memory Test Logic (MTL), located on the NCU, and does not involve the heads in any way. Therefore, it is not possible to redirect this function to a head, or I/O, as was possible on other C series architectures.

It should be possible to initialize any combination and configuration of memory, which includes 1 thru 8 NMB's. If this function fails any configuration, even intermittently, a failure should be suspected.

The most common cause of mminit failures involve the send portion (XS0 and XS1) of the crossbar. By use of the utility nmb_errs, it should be quite obvious whether the failure is address, or data oriented.

It might be helpful to understand that mminit acts very much like a simple memory diagnostic. This logic is further tested by means of subtests 380 and 385 of mem4000.

Failures of mminit indicating odd bank 15 of the last NMB installed are generally evidence of the need for a sysreset -l 2. The mminit utility will frequently fail in this manner if initall, or a sysreset -l 2 has not been accomplished. A standard sysreset will not usually be sufficient.

16.* NMB Failure Precautions

It is necessary to physically remove a disabled NMB before trying to boot the C3800. It is not sufficient to simply remove power from a defective memory board and otherwise leave the board installed.

If the NMB is left inserted in the slot, memory failures will be encountered when trying to boot to multiuser. It is also possible to encounter failures with diagnostics.

This does not apply to Sp's and Vp's. Simply removing power when disabling a head is sufficient to remove the failed head from the system.

C3800 Troubleshooting Guide

17.* Crossbar/NCU Power Removal Warning

If, for any reason, it is necessary to power the NCU and/or XBAR down, then it will be necessary to perform a sys_shutdown and diaginit before attempting to boot the system. Just relinking the NCU and crossbar with diaginit will not be sufficient to restore scannability of the entire system. In many cases it will even be impossible to mminit if the entire system is not reinitialized before continuing.

18.* NSP Revision History

In an effort to answer the most frequently asked question related to progress on fixing the known problems with C3800's, this information is being made available. Please, bear in mind that these boards cannot be ordered based on these upgrades. This is for information purposes only. Below is a list of NSP revision levels and the problem intended to address.

C.1

Another attempt to fix nmul failures, this time tying clk* to vref and moving clk out by ~400ps. Failing symptom is byte 4 ymux parity error on nmul, suspected to be a setup time problem on the nmul. Added to other function units as well. Also removes X.5 (function unit vref adjustment ecn).

D.0

Removes rev B completely. Installs three 0.01uF capacitors on pins (7 (Vref), 11 (Vcc), and 26 (Vcca)) to the lid (vtt) of the purge rams to reduce noise. Was wire revF xecn X.11.

Deletes clock delay wires of rev C and replaces them with 11 inch wires. Delays clocks to the function units an additional ins. Was wire revF xecn X.10.

Ecn to keep the branch history and referenced PRAMS from being able to cause hard errors, since they are non-destructive if wrong. Was wire revF xecn X.12.

Disable OLE on the dcache and ptecache prams by pulling it high. This helps the problem we were seeing where ring data does not match ram data on a pram failure, at first thought to be an access time problem. Was wire revF xecn X.7.

Purge rams addressed in this release:

BR_HIST_L BR_HIST_U BR_HIST_XL BR_HIST_XU PRD_REFO PRD_REF1

E.0

Turns off parity checking on purge RAM modify bits.

93/02/22
07:51:00

40

C3800 Troubleshooting Guide

F.0

Hardware support for project to coredump processes that pull hard errors instead of crashing the complex. Need to guarantee that we don't hang the ncu because of pending traps; that we don't change the cir, tid, ring, idle status, and we keep interrupt state; and lastly we can't make requests to memory.

G.0

Disable parity checking on purge RAM lookahead valid bits.

H.0

Fixes assy revF problem where is used a register (spare8) that is held by some stram wr_par_err's. We now use register (stop_cntr) and the old stop_cntr is now (spare0).

J.0

Power reference modifications for NMUL problems

K.0

Eliminates SA1_Miss3_Stop_IN failure that accompanies other crash messages.

19.* C3800 Hard Error List

The following is a list of all possible hard errors associated with the C3800:

MB_BOARD:

mb_bcga_ise_cycle,	mb_even_illegal_row_addr,
mb_bcga_ise_hi_addr,	mb_ise_addr_perr,
mb_bcga_ise_low_addr,	mb_ise_data_perr,
mb_bcga_ise_mid_addr,	mb_ise_zone_perr,
mb_bcga_iso_cycle,	mb_iso_addr_perr,
mb_bcga_iso_hi_addr,	mb_iso_data_perr,
mb_bcga_iso_low_addr,	mb_iso_zone_perr,
mb_bcga_iso_mid_addr,	mb_odd_bank_ctl_err,
mb_even_bank_ctl_err,	mb_odd_bank_cycl_err,
mb_even_bank_cycl_err,	mb_odd_bank_cyc2_err,
mb_even_bank_cyc2_err,	mb_odd_bank_cyc3_err,
mb_even_bank_cyc3_err,	mb_odd_bank_perr,
mb_even_bank_perr,	mb_odd_illegal_dram_addr,
mb_even_illegal_dram_addr,	mb_odd_illegal_row_addr;

VP_BOARD:

a_vm.uir_par_err,	ndiv5.xbus_par_err,	nmul0.xbus_par_err,
l_vm.uir_par_err,	ndiv5.ybus_par_err,	nmul0.ybus_par_err,
m_vm.uir_par_err,	nfad_a.xbus_par_err,	nmul1.xbus_par_err,
ndiv.xbus_par_err,	nfad_a.ybus_par_err,	nmul1.ybus_par_err,
ndiv.ybus_par_err,	nfad_m.xbus_par_err,	nmul2.xbus_par_err,
ndiv0.xbus_par_err,	nfad_m.ybus_par_err,	nmul2.ybus_par_err,

C3800 Troubleshooting Guide

ndiv0.ybus_par_err,	nib_par_err,	nmul3.xbus_par_err,
ndiv1.xbus_par_err,	nis0.is_par_err,	nmul3.ybus_par_err,
ndiv1.ybus_par_err,	nis1.is_par_err,	nvd.vd_par_err,
ndiv2.xbus_par_err,	nmisc_a.xbus_par_err,	nvp_hard_err,
ndiv2.ybus_par_err,	nmisc_a.ybus_par_err,	rsit_par_err,
ndiv3.xbus_par_err,	nmisc_m.xbus_par_err,	
ndiv3.ybus_par_err,	nmisc_m.ybus_par_err,	
ndiv4.xbus_par_err,	nmul.xbus_par_err,	
ndiv4.ybus_par_err,	nmul.ybus_par_err;	

SP_BOARD:

dcd0.wr_par_err,	nfad.xbus_par_err,	nrc0.rctrl_par_err,
dcd1.wr_par_err,	nfad.ybus_par_err,	nrc1.edata_que_par_err,
dct0.wr_par_err,	niad0.mxi_par_err,	nrc1.odata_que_par_err,
dctl.wr_par_err,	niad0.ybus_par_err,	nrc1.rctrl_par_err,
dcu0.wr_par_err,	nrc.xre_stop_in,	nrc.xro_stop_in,
niad1.mxi_par_err,	nrc2.edata_que_par_err,	dcu1.wr_par_err,
niad1.ybus_par_err,	nrc2.odata_que_par_err,	icd.wr_par_err,
nmisc.xbus_par_err,	nrc2.rctrl_par_err,	ict.wr_par_err,
nmisc.ybus_par_err,	nrfa.extdata_par_err,	lat.wr_par_err,
nmul.xbus_par_err,	nrfa0.bbush_par_err,	nag0.displ_par_err,
nmul.ybus_par_err,	nrfa0.cbush_par_err,	nag0.lxaq_par_err,
npa0.ctag_par_err,	nrfa0.displ_par_err,	nag0.uirl_par_err,
npa0.ctval_par_err,	nrfa0.nrfa_wcs_par_err,	nag0.updq_par_err,
npa0.cval_par_err,	nrfa1.bbush_par_err,	nag0.vxaq_par_err,
npa0.pdata_par_err,	nrfa1.cbush_par_err,	nag0.zbus_par_err,
npa0.ptag_par_err,	nrfa1.displ_par_err,	nag1.displ_par_err,
npa0.ptval_par_err,	nrfa1.nrfa_wcs_par_err,	nag1.lxaq_par_err,
npa0.pval_par_err,	nrfa2.bbush_par_err,	nag1.uirl_par_err,
npal.ctag_par_err,	nrfa2.cbush_par_err,	nag1.updq_par_err,
npal.ctval_par_err,	nrfa2.displ_par_err,	nag1.vxaq_par_err,
npal.cval_par_err,	nrfa2.nrfa_wcs_par_err,	nag1.zbus_par_err,
npal.pdata_par_err,	nrfa3.bbush_par_err,	ndc.uirl_par_err,
npal.ptag_par_err,	nrfa3.cbush_par_err,	ndiv.xbus_par_err,
npal.ptval_par_err,	nrfa3.displ_par_err,	ndiv.ybus_par_err,
npal.pval_par_err,	nrfa3.nrfa_wcs_par_err,	ndp.yuvc_par_err,
npar.br_que_par_err,	nsp_hard_error_script,	ndp0.cupd_par_err,
npar.inst_tag_par_err,	nus.nus_wcs_par_err,	ndp0.pmod_par_err,
npar.inst_valid_par_err,	nus.srv_rd_par_err,	ndp0.pref_par_err,
npar.look_tag_par_err,	pted0.wr_par_err,	ndp1.cupd_par_err,
npar.look_valid_par_err,	pted1.wr_par_err,	ndp1.pmod_par_err,
npar.que_par_err,	ptet0.wr_par_err,	ndp1.pref_par_err,
npsw.npsw_wcs_par_err,	ptet1.wr_par_err,	ndp2.cupd_par_err,
nrc.par_err_stop_in,	ndp2.pmod_par_err,	nrc0.edata_que_par_err,
ndp2.pref_par_err,	nrc0.odata_que_par_err,	serd.wr_par_err,
nrc.eybus_stop_in,	nrc.oybus_stop_in,	nrc.sal_miss3_stop_in,
npsw.statq_hard_err,	npsw.unicode_pulled_hard_err;	

IA_BOARD:

ia_nxi_hard_error,	ia_rdq_flag_pe,	ia_xds_hdr_pe,	ia_pi_hard_error,
ia_rdq_wrt_pe,	ia_xds_wde_pe,	ia_rd_par_err_e,	ia_wdq_fflag_pe,
ia_xds_wdo_pe,	ia_cds_rd_pe,	ia_rd_par_err_o,	ia_wdq_wrt_pe;

CU_BOARD:

cu_addr_par_err.0,	cu_addr_par_err.1,	cu_addr_par_err.3.2,
cu_lckb_par_err,	cu_ndat0_harderr,	cu_ndat1_harderr,
cu_ram_par_err,	cu_trp_harderr,	cu_wr_dat_par_err;

C3800 Troubleshooting Guide

XSO_BOARD:

xs0.bad_send_par_err, xs0.lone_hard_err, xs0.pcm_err_a,
xs0.pcm_err_b, xs0.send_par_err_mb0, xs0.send_par_err_mb1,
xs0.send_par_err_mb2, xs0.send_par_err_mb3, xs0.send_par_err_mb4,
xs0.send_par_err_mb5, xs0.send_par_err_mb6, xs0.send_par_err_mb7,
xs0.send_par_err_ncu;

XS1_BOARD:

xs1.bad_send_par_err, xs1.lone_hard_err, xs1.send_par_err_mb0,
xs1.send_par_err_mb1, xs1.send_par_err_mb2, xs1.send_par_err_mb3,
xs1.send_par_err_mb4, xs1.send_par_err_mb5, xs1.send_par_err_mb6,
xs1.send_par_err_mb7, xs1.send_par_err_ncu;

XRT_BOARD

rctl.bad_req_mb0, rctl.bad_req_mb1, rctl.bad_req_mb2,
rctl.bad_req_mb3, rctl.bad_req_mb4, rctl.bad_req_mb5,
rctl.bad_req_mb6, rctl.bad_req_mb7, rctl.bad_req_ncu,
rctl.cr_accel_err, rxbr0.overflow, rxbr0.underflow, rxbr1.overflow,
rxbr1.underflow, rxbr2.overflow, rxbr2.underflow, rxbr3.overflow,
rxbr3.underflow, rxbr4.overflow, rxbr4.underflow, xrt.bad_rd_rdy_mb0,
xrt.bad_rd_rdy_mb1, xrt.bad_rd_rdy_mb2, xrt.bad_rd_rdy_mb3,
xrt.bad_rd_rdy_mb4, xrt.bad_rd_rdy_mb5, xrt.bad_rd_rdy_mb6,
xrt.bad_rd_rdy_mb7, xrt.bad_rd_rdy_ncu, xrt.bubble_error,
xrt.no_rd_rdy_mb0, xrt.no_rd_rdy_mb1, xrt.no_rd_rdy_mb2,
xrt.no_rd_rdy_mb3, xrt.no_rd_rdy_mb4, xrt.no_rd_rdy_mb5,
xrt.no_rd_rdy_mb6, xrt.no_rd_rdy_mb7, xrt.no_rd_rdy_ncu,
xrt.rtn_par_err_nia, xrt.rtn_par_err_sp0, xrt.rtn_par_err_sp1,
xrt.rtn_par_err_sp2, xrt.rtn_par_err_sp3, xrt.rtn_par_err_sp4,
xrt.rtn_par_err_sp5, xrt.rtn_par_err_sp6, xrt.rtn_par_err_sp7;

20.* xsysconfig reminder

Please remember that it is possible to change C3800 hardware configurations by means of the xsysconfig feature. This will allow both NMB's and heads to be removed and reinstalled from boot configurations without having to power down the head and memory boards. This will save wear and tear on the components themselves.

When using xsysconfig to modify configurations, it is possible to encounter memory timing errors and memory faults at boot. Ignore these and continue with the boot. These failures are not fatal and should cause no problems after the system is booted.

21.* Powered Down Bay Detection

As a suggestion, it might be beneficial, for troubleshooting purposes, to cut 2 to 3 inch strips of mag tape and tape to the top of C3 bays. In this way, any powered down bays can be spotted very quickly. With the noise, usually associated with computer rooms it is near impossible to tell if a single bay has powered off. This can result in serious delays in identifying the cause of a crash or failed boot.

Of course this doesn't help in the case of a remote session, but individuals are generally more cautious and observant when dialed in.

C3800 Troubleshooting Guide

22.* Removing Defective Boards

It is recommended that when removing individual boards from a head on a C3800, that the accompanying board be backed out of the backplane. For instance, if removing a NSP to return to Dallas, then back the NVP out as well.

The reason for this is that some interaction with the remaining board, even though powered off, has been associated with intermittent spu4000 failures and in isolated instances, even system crashes.

23.* C3800 wredc Parity Errors at Upper Clock

It has been discovered that memory wredc parity errors will occur when running a C3800 at upper clock. The failure will always indicate BANK 0o and can occur on any NMB.

An example of the error can be seen below. As indicated this error should be the only hard error reported at the time of the failure.

The cause of this problem is being investigated, but until it is solved, please bear this in mind. The clock frequency at the time of the crash can be confirmed from the output of the hard logger, directly after the cop is completed.

Hard error Summary

HARDERROR #	BOARD TYPE	PORT/SIDE	EXTRACTOR
0	MB_TYPE	0	mb_odd_bank_wredc_par_err

ODD side 'bank_wredc_perr': PARITY error detected in MB6

WREDC PARITY error detected by BCGA 0 in bank 0 (=> BANK_0o) during a READ.

1	mbs6:bc[0].sys.bctl0_merr_pe		
1	mbs6:nmc0o_rerr		
1	mbs6:nmc0o_ria_ecc_check1		
10480000	mbs6:nmc0o_data		
f	mbs6:nmc0o_ecc (parity)		
1	mbs6:bc[0].bcga_log.bctl0_cycle		
e7e6e0	mbs6:bc0_bank0_log_addr		(phys_addr is 73f3700)

24.* Undetected NMB deallocation

During periods of troubleshooting, or system reconfiguration, it is possible for memory boards to actually drop off line. This means that the NMB remains powered up and for all practical purposes, is physically in the configuration, but for purposes of use it is not available.

93/02/22
07:51:00

(trngps)(mikey:throg Job: vol. 1 Date: Mon Feb 22 07:50:58 1993)

psfilter.in.23447

42

C3800 Troubleshooting Guide

The symptoms will be that the NMB shows up with no problems after the diaginit is executed, but mminit does not run on the NMB, or after booting the system, the NMB is unavailable.

This problem, though somewhat frustrating, is easily solved.

By using the sys_config window it is quite quickly seen that there are two sections. The top of the display indicates components physically installed in the system and the bottom half displays components that are not available to OS. It will generally be found that the offending NMB's are not available for OS. To solve the problem, it is only necessary to click on these entries with the mouse and after an install, this will restore all available components to the configuration.

25.* C3800 New Boards

This is intended to explain some new board respins and board revisions on the 3800 that have been reported in other documentation. This is not intended to indicate the availability of these components, but only to inform the field of changes to be applied in the future.

XCL2

This respin will allow for the logging of soft errors with new purge rams available on the 8247 rev H NSP. This capability will not be available on current NSP's and will not be available before Q4.

There will be no retrofit in the field.

XCL1 and XCL2 will be interchangeable.

NMB2/BCGA3/60ns DRAM/NMBPP2

Increases memory performance and allows use of 1/4 populated NMC. Allows 128Mb memory capacity without 1Mb Dram.

NMB2 and BCGA3 required for performance increase.

When using NMB2's with NMB1's no performance increase possible.

New parts cannot be used on existing NMB's. No retrofit for the field. Use current NMB's until exhausted.

Not available before 93.

XDS2 Gate Arrays

Resident on the NIA. Allows for interleaving to be tuned for faster performance when installed with 8247 and 9247 NSP's.

No field upgrade. Not available before Q4

Referred to as NIA2.

C3800 Troubleshooting Guide

8247 NSP

This board is a total respin with the respun gate arrays (NDP, NUS2, NPA) which is to allow the handling of purge ram soft errors.

This board will work in conjunction with the XCL2.

Available in Q4. NO field upgrade.

9247 NSP

This is a 6247 NSP with the new gate arrays, as indicated above, installed. However, the soft error handling is still kludged on the board itself and therefore this board will not function with XCL2's.

This NSP will generate soft errors for all purge rams except the instr_valid.

No field upgrade.

26.* Restarting a C3800 After a Crash

It has been found that in order to cleanup properly, after a crash, it is generally necessary to issue 3 consecutive cleanup commands, followed by a "sysreset -s". This is required to avoid problems with automated scripts, such as the auto-reboot script.

The procedure would appear as follows:

```
osclean
cleanup
cleanup
cleanup
sysreset -s
```

This should be taken into account any time it is necessary to recover after a system crash.

27.* Distinguishing STRAM and Purge Ram Failures

The following error will accompany any STRAM, or purge ram failure that is reported by the C3800 hard_logger. This error should be ignored, unless it occurs by itself. The other hard errors should be considered the root cause of the crash.

In addition to this "tag along" error, it will also be noticed that hard_err2 will generally contain a 00000041, which indicates that both even and odd crossbar return boards are reporting the failure. As can be seen below the erroneous failure is reported by the Return Control gate array.

The rev K NSP will eliminate this "tag along" nuisance.

C3800 Troubleshooting Guide

```

-----
HARDERROR #   BOARD TYPE   PORT/SIDE   EXTRACTOR
-----
      0         SP_TYPE       1           sp_nrc.sal_miss3_stop_in
-----
spl:nrc0.par_err_stop_in = 1 \
spl:nrc1.par_err_stop_in = 1 | These should all have the same value
spl:nrc2.par_err_stop_in = 1 /
Parity error script for
sal_miss1 bus received by NRC0, NRC1, and NRC2 gate arrays.
Register name      Ring value      Board signal name
-----
sal_miss3<>      00 06 cc 68      SA1_MISS1<31..3>;SA1_ADDR2<2..0>
sal_miss3<>      1  1  1  1*      SA1_ADDR2_PAR<0..3>
* indicates parity error
+++>

```

```

<Wed Jul 29 13:32:07 1992> logmsg:../logmsg.c:56
Hard Error Message from Extractor (DiagER349): logmsg: General purpose event
Source: sp_nrc_sal_miss3_stop_in
@extractor=sp_nrc_sal_miss3_stop_in @board_type=sp @port=1
****

```

Another way to determine if it would be worthwhile to try swapping nvp's around would be to examine the nvrf.out.<date>.Z file in /sst. If it has a line that has a parity error, such as,

```

v1[126]=00000000bfa3a8ba
v1[127]=00000000be6321d0
v2[000]=000000404043116a PARITY ERROR:
Expected: byte c7, nibble 87; Actual: byte cf, nibble 8f
v2[001]=0000000040fcbb71
v2[002]=0000000040216b32

```

Then the failure is a real nvrf2 error and no amount of swapping will fix it.

However if there isn't a line such as this in the file then the error is not within the nvrf ram and may be a interconnect problem between the nsp and nvp, and swapping nvp's may be in order.

It should be noted that just because there isn't an error in the ram doesn't mean that it couldn't possibly be a nvrf2 error, since the error may have been written over before clock were stopped, but it is an indication.

28.* Memory Configuration related Crashes

It should be understood that it is possible to generate a crash on a 3800 when attempting to execute code on a system with more cpu's installed than memory boards. This can occur because of problems with Hitachi DRAMS and will occur on an individual memory board if, even one of these DRAM's is installed on a NMC.

This symptom has, also, been seen on systems running odd numbers of NMB's. For instance, running 3, 5 and 7 boards can encounter this particular symptom as well.

C3800 Troubleshooting Guide

The crash that will be encountered is a rdedc hard error. An example of this will appear below:

```

-----
HARDERROR #   BOARD TYPE   PORT/SIDE   EXTRACTOR
-----
      0         MB_TYPE       4           mb_odd_bank_rdedc_multi_bit_err
-----
ODD side 'bank_rdedc_multi_bit_err' detected in RDEDCE of MB4
-----
MULTI-BIT error in BCGA BANK3 ( => BANK 8o) detected by RDEDCE
during READ operation.

```

This error will be encountered most often when executing code on multiple cpu's and one memory module on line.

29.* Memory Errors

As reported in a previous tech bulletin, the error shown below can be expected to occur on NMB's with Hitach Drams installed and the system configured with more processors than memory boards installed.

The problem has been identified as a timing problem and is fixed on 1246 NMB's assembly rev B. The solution is to gate the data through an additional register, which is enough to overcome the timing difficulty.

The failure is not severe enough to justify a field upgrade, but all NMB's returned to Dallas will be upgraded during the repair process.

```

-----
HARDERROR #   BOARD TYPE   PORT/SIDE   EXTRACTOR
-----
      0         MB_TYPE       4           mb_odd_bank_rdedc_multi_bit_err
-----

```

ODD side 'bank_rdedc_multi_bit_err' detected in RDEDCE of MB4

MULTI-BIT error in BCGA BANK3 (=> BANK 8o) detected by RDEDCE during READ operation.

30.* Identifying STRAM and PRAM Failures

There has been some confusion in identifying stram and pram failures. In particular, because of the similar extractor names there has been some difficulty in differentiating between pram and stram failures. This document is intended to clarify some of this confusion.

Although these are very basic examples, they should illustrate the difference between the two components.

As STRAM's are utilized in system cache's, tags and scratch rams, therefore these failures will involve data fields ranging from 16 to 64 bits, as can be seen in examples 1 and 3 below:

C3800 Troubleshooting Guide

Purge rams, on the other hand are used for cache housekeeping, updates and validity. So the data fields involved in this are much smaller.

As can be seen from these two examples, both hard errors indicate various size data fields.

The purge ram failures are displayed in examples 2, 4 and 5. As can be seen from these examples, the purge rams consist of 2 bit data fields, in this case <1..0>.

EXAMPLE 1

```
-----
HARDERROR #   BOARD TYPE   PORT/SIDE   EXTRACTOR
-----
1             SP_TYPE      0           sp_npar.inst_tag_par_err
-----
```

```
sp0:npar.inst_tag_par_err = 01
Parity error script for parser_inst_tag received by NPAR gate array
Ram Address = 7b5 INST_CACHE_ADDR<13..3>
Ram value      Board signal name
```

```
-----
Ring  00 01 80  PARSE INST_TAG<31..14>::zero<13..8>
      1  1* 0   PARSE INST_TAG<32..34>
Ram   00 09 80
      1  1  0
```

* indicates parity error

```
+++>
<Thu Jun 25 22:06:38 1992> logmsg:../logmsg.c:56
Hard Error Message from Extractor (DiagER349):logmsg:General purpose event
Source: sp_npar_inst_tag_par_err
@extractor=sp_npar_inst_tag_par_err @board_type=sp @port=0
****
```

EXAMPLE 2

```
-----
HARDERROR #   BOARD TYPE   PORT/SIDE   EXTRACTOR
-----
1             SP_TYPE      2           sp_npa0.pval_par_err
-----
```

```
sp2:npa0.pval_par_err = 1
Parity error script for prd_val0 received by NPA0 gate array
Ram Address = 636 SA0<22..12>
Source      Value      Board signal name
```

```
-----
Ring  1*   PRD_VAL0<1..0>
Ram   1*
```

* indicates parity error

+++>

C3800 Troubleshooting Guide

EXAMPLE 3

```
-----
HARDERROR #   BOARD TYPE   PORT/SIDE   EXTRACTOR
-----
1             SP_TYPE      6           sp_npar.que_par_err
-----
```

```
sp6:npar.que_par_err = 01
Parity error script for icac_data_out received by NPAR gate array
Ram Address = 781 INST_CACHE_ADDR<13..3>
Source      Value      Board signal name
```

```
-----
Ring  14 89 ff fc 36 39 7f a8  ICAC_DATA_OUT<63..0>
      1  0 1  1  1  1  1*  0  ICAC_DATA_OUT<64..71>
Ram   14 89 ff fc 36 39 ff a8
      1  0 1  1  1  1  1  0
```

* indicates parity error

+++>

EXAMPLE 4

```
-----
HARDERROR #   BOARD TYPE   PORT/SIDE   EXTRACTOR
-----
1             SP_TYPE      0           sp_npai.ctval_par_err
-----
```

```
sp0:npai.ctval_par_err = 1
Parity error script for crd_tval1 received by NPA1 gate array
Ram Address = 49d SA1<13..3>
Source      Tvalue     Board signal name
```

```
-----
Ring  1*   CRD_TVAL1<1..0>
Ram   1*
```

* indicates parity error

+++>

```
<Wed Jul 22 09:54:17 1992> logmsg:../logmsg.c:56
Hard Error Message from Extractor (DiagER349):logmsg:General purpose event
Source: sp_npai_ctval_par_err
@extractor=sp_npai_ctval_par_err @board_type=sp @port=0
****
```

EXAMPLE 5

```
-----
HARDERROR #   BOARD TYPE   PORT/SIDE   EXTRACTOR
-----
1             SP_TYPE      1           sp_ndpl.pmod_par_err
-----
```

```
sp1:ndpl.pmod_par_err = 1
Parity error script for prd_mod1 received by NDP1 gate array
Ram Address = 62d SA1<22..12>
Source      Value      Board signal name
```

```
-----
Ring  2*   PRD_MOD1<1..0>
Ram   2*
```

* indicates parity error

+++>

93/02/22
07:51:00

(trngps)(mikey:throg Job: vol.8 Date: Mon Feb 22 07:50:58 1993)

45

psfilter.in.23447

C3800 Troubleshooting Guide

<Wed Aug 5 11:38:51 1992> logmsg:../logmsg.c:56
Hard Error Message from Extractor (DiagER349):logmsg:General purpose event
Source: sp_ndpl_pmod_par_err
@extractor=sp_ndpl_pmod_par_err @board_type=sp @port=1

31.* NVRF2 Failures

It should be understood that the "NVRF2" failure is not the only failures related to the yuvc register. In addition there is only one of these type failures that is a known problem and should be ignored. The other failures are legitimate and should be addressed when they occur.

The yuvc register is the source of parity checking for 4 data busses off of the 3 NDP (Data Path Gate arrays). The 3 NDP's are partitioned in such a way as to allow byte rotation without involving multiple gate arrays. This is accomplished by allowing each gate array to handle 3 bits of each byte. For each byte 1 gate array will handle 2 bits of data and 1 bit of parity for that byte.

The four busses involved in the yuvc are:

CRD Data Cache Read Data
UPD Data Cache Update
Ybus The primary SP external Data bus
VX Vector to Scalar Data bus

The NVRF2 failure involves, only, the VX bus and is generally originates on the Vector board and is finally detected at the yuvc. This failure can sometimes be addressed by rotating vector boards between SP's, but this is not recommended unless encountering multiple VX data parity errors on the same head. An example of this "known" failure appears below:

Hard error Summary

```
-----  
HARDERROR # BOARD TYPE PORT/SIDE EXTRACTOR  
-----  
0 SP_TYPE 0 sp_ndp.yuvc_vx_err  
-----  
sp0:ndp0.yuvc_par_err = 4 \  
sp0:ndp1.yuvc_par_err = 4 | These should all have the same value  
sp0:ndp2.yuvc_par_err = 4 /  
Parity error script for Vx bus received by  
NDP0, NDP1, and NDP2 gate arrays.  
Register name Ring value Board signal name  
-----  
vx<> 00 00 80 00 00 00 4e 43 VX_DATA<63..0>  
vx<> 1 1 1* 1 1 1 1 0 VX_PAR<0..7>  
* indicates parity error  
+++>
```

C3800 Troubleshooting Guide

<Tue Jul 28 20:07:56 1992> logmsg:../logmsg.c:56
Hard Error Message from Extractor (DiagER349):logmsg:General purpose event
Source: sp_ndp_yuvc_vx_err
@extractor=sp_ndp_yuvc_vx_err @board_type=sp @port=0

If the failure involves one of the other 3 busses and is not a VX data parity error than the problem is Scalar in origin and indicates a failure. These faults should be addressed. A couple of examples of these failures can be seen below:

As can be seen by the next example, this error occurs on the UPD bus.

The next error is indicated on the CRD bus and is a STRAM failure. As with the UPD par_err, this problem should not be overlooked, as it is a more serious issue.

Hard error Summary

```
-----  
HARDERROR # BOARD TYPE PORT/SIDE EXTRACTOR  
-----  
2 SP_TYPE 1 sp_ndp.yuvc_upd_err  
-----  
sp1:ndp0.yuvc_par_err = 1 \  
sp1:ndp1.yuvc_par_err = 1 | These should all have the same value  
sp1:ndp2.yuvc_par_err = 1 /  
Parity error script for Upd bus received by  
NDP0, NDP1, and NDP2 gate arrays.  
Register name Ring value Board signal name  
-----  
upd<> 00 00 00 00 00 00 00 20 UPD_DATA<63..0>  
upd<> 1 1 1 1 1 1 1 1* UPD_PAR<0..7>  
* indicates parity error  
+++>  
<Sat Jun 20 12:22:10 1992> logmsg:../logmsg.c:56  
Hard Error Message from Extractor (DiagER349):logmsg:General purpose event  
Source: sp_ndp_yuvc_upd_err  
@extractor=sp_ndp_yuvc_upd_err @board_type=sp @port=1  
****
```

Hard error Summary

```
-----  
HARDERROR # BOARD TYPE PORT/SIDE EXTRACTOR  
-----  
0 SP_TYPE 1 ndp.yuvc_crd_err  
-----  
sp1:ndp0.yuvc_par_err = 1 \  
sp1:ndp1.yuvc_par_err = 1 | These should all have the same value  
sp1:ndp2.yuvc_par_err = 1 /  
Parity error script for Ybus, Upd, Vx, and Crd buses received by  
NDP0, NDP1, and NDP2 gate arrays.  
Register name Ring value Board signal name
```

93/02/22
07:51:00

(trngps)(mikey:throg Job: vol.8 Date: Mon Feb 22 07:50:58 1993)

psfilter.in.23447

46

C3800 Troubleshooting Guide

```
-----  
ybus<> 00 00 00 70 0f 22 c0 00 YBUS_DATA<63..0>  
ybus<> 1 1 1 0 1 1 1 1 YBUS_PAR<0..7>  
upd<> 20 00 0b 10 20 00 3b 17 UPD_DATA<63..0>  
upd<> 0 1 0 0 0 1 0 1 UPD_PAR<0..7>  
vx<> 00 00 00 00 00 00 80 00 VX_DATA<63..0>  
vx<> 1 1 1 1 1 1 1 1 VX_PAR<0..7>  
crd<> 00 00 00 01 22 f0 35 c6 CRD_DATA0<31..0>  
/CRD_DATA1<31..0>  
crd<> 1 1 1 0 1 1 1 *1  
CRD_DATA0<32..35>/CRD_DATA1<32..35>  
* indicates parity error
```

As mentioned in previous bulletins, the NVRF2 failure is being pursued and will be solved. But, until that time it is important to be able to distinguish this failure from other errors, reported by the same mechanism.

32.* New Keyswitch

A new keyswitch will soon be available for the 3800 systems. It should be available to the field by the middle of November and will be in new builds in the first week of November.

The part number for this keyswitch is 500-000542-200.

The prerequisite for this switch is 3.1 Diagnostics. The old keyswitch software will continue as default until the new switch is installed. Both versions of the software will be loaded as shown below:

```
xsfp.old_switch/xsfp.rc.old_switch  
xsfp.new_switch/xsfp.rc.new_switch
```

When the new switch is installed, the new software can be initiated with the following script:

```
install_new_keyswitch
```

If necessary, the old switch software can be initiated by:

```
install_old_keyswitch
```

The features of the new keyswitch is as follow:

1. Positions are Off-Remote-Local-Secure (was Off-Local-Secure-Remote).
2. xsfp now runs sys_shutdown in the Off position (was a hard bpc reset).
3. Key can be removed in the Off or Secure positions.

C3800 Troubleshooting Guide

xsfp now waits 1 second after the last keyswitch interrupt before processing interrupt.

The procedure for installation of the new keyswitch is as follows:

1. Run sys_shutdown
2. Replace keyswitch
3. Run install_new_keyswitch
4. When prompted, reboot SPU

33.* NCU related error: init_commregs

In a previous release of the tech bulletins, the failure indication below was mentioned. In this bulletin, which is also in article 33 of chapter 3, in the C3800 Troubleshooting Hints.

```
init_commregs: error clearing comm reg 0xc lock  
init_commregs: error clearing comm reg 0xd lock  
init_commregs: error clearing comm reg 0xe lock  
init_commregs: error clearing comm reg 0xf lock
```

Again, this failure will only be seen at boot.

The previous information indicated that the mechanism of failure was unknown, but the solution was to reload diagnostics.

It is now known that the failure mechanism is a jammed data base and can be cleared with the following procedure:

```
sys_shutdown  
cd /diag/db  
rm cdb.db* cdb.map* rb_*  
kill_by_name cdb_startup
```

This procedure will remove the current database files, kill the current cdbserver and rbserver processes and restart them.

With everything that is now known, this should clear the problem with a minimum of trouble and effort.

93/02/22
07:51:00

(trngps)(mikey:throg Job: vol.8 Date: Mon Feb 22 07:50:58 1993)

psfilter.in.23447

47

C3800 Troubleshooting Guide

34.* Potential epont Problem on C3800

There is growing evidence to suggest that the epont connectors, used in the C3800, may be experiencing stress, due to vibration, heat and other factors that may contribute to connector associated failures.

There is a possibility that the epont retaining screws may back out over time and break the seal, allowing contaminants to work between the fuji poly and backplane connection. This situation can lead to intermittent and widely scattered crossbar related failures.

Because this possibility exists, it is recommended that all eponts be checked to verify proper torque as soon as practical for each system.

If any loose connectors are discovered, it is requested that this information be relayed to hardware support.

This situation is, not yet, deemed critical and should be considered a preventive maintenance item.

35.* Updating 3800 Data Base

As with other CONVEX systems, it is possible to add a new board and part number to the data base, on a C3800. The approach is somewhat different though.

The file to be edited is /diag/db/part_numbers. It is very similar to the file DB_cop, on a C2. But in order to get the part actually included in the data base, it is necessary to execute the command "add_parts". This will take any new entries in the part_numbers file and move them to the configuration data base. If no changes are necessary then "add_parts" will respond with "no new part numbers were found".

If the "add_parts" command is not executed, then the board will not be available until the SPU is rebooted. At a reboot the new part number will be added to the data base.

An example of the part_numbers file is shown below:

```
410-001149/viop/1&1:A-A,ZZ-ZZ;&
410-001269/tli/1&1:A-A,ZZ-ZZ;&
410-001228/idc/1&1:A-A,ZZ-ZZ;&
410-001328/hpi/1&1:A-A,ZZ-ZZ;&
410-001132/hsp/1&1:A-A,ZZ-ZZ;&
410-002132/hsp/1&1:A-A,ZZ-ZZ;&
411-000243/mc/0&
411-001243/mc/0&
411-002243/mc/0&
411-004243/mc/0&
411-005243/mc/0&
411-006243/mc/0&
500-001252/xbp/0&
500-001253/ccubpl/0&
500-001254/ccubpr/0&
```

C3800 Troubleshooting Guide

```
411-001272/wi/0&
500-001251/iobp/0&
500-003251/iobp/0&
500-001286/cbpl/0&
500-002286/cbpl/0&
500-001287/cbpr/0&
500-002287/cbpr/0&
410-002149/viop/1&1:A-A,ZZ-AA;&
416-001244/ia/1&0:A-A,ZZ-ZZ;&
416-003244/ia/1&1:B-0,ZZ-ZZ;&
416-004244/ia/1&2:B-0,ZZ-ZZ;&
416-001246/mb/1&0:A-A,ZZ-ZZ;&
416-003246/mb/1&1:A-A,ZZ-ZZ;&
```

36.* ECC Timed Backoff Error Message

The memory messages, shown below, occur as a result of the system trying to throttle the single bit memory errors, to avoid swamping the soft log.

The total explanation follows:

Single-bit memory error data is stored in the CDB. Single-bit memory errors are isolated to the memory chip level. A count of total soft errors for each failed memory chip is maintained. By default, errintd will store a maximum of 60 memory chip entries in the CDB.

Once the number of total memory chip failures reaches 75% capacity and a burst of errors occur (e.g., at a rate of 1 every 10 seconds), the logging of new chips in error is throttled, or governed, to prevent the log from immediately reaching its capacity. Whenever throttling occurs, a message is written to the console.

```
Loading vmunix
mm_sniff: sniff rate: 32.14 MB/day (7.90 days/pass)
errintd: Soft ECC timed backoff(1): 0.750 secs.
errintd: Soft ECC timed backoff(2): 0.750 secs.
errintd: Soft ECC timed backoff(3): 0.750 secs.
errintd: Soft ECC timed backoff(4): 1.000 secs.
errintd: Soft ECC timed backoff(5): 2.000 secs.
errintd: Soft ECC timed backoff(6): 5.000 secs.
errintd: Soft ECC timed backoff(7): 20.000 secs.
```

These errors will frequently be reported at boot, when a memory board has been powered off and left installed. As previous tech bulletins have indicated, a powered down memory board must be physically removed from the system.

In addition, a nmc failure can result in the same output.

93/02/22
07:51:00

(trngps)(mikey:throg Job: vol.8 Date: Mon Feb 22 07:50:58 1993)

48

psfilter.in.23447

C3800 Troubleshooting Guide

37.* Crashdump Validity

It has been discovered that many of the C3800 SPU based utilities and interactive tests invalidate the validity of a crashdump. For this reason it is recommended that no SPU based utilities be considered if it is felt that a crashdump will be necessary to pursue the problem.

These utilities would include:

rslog	xbar_err	pb_walk	idc_con
syscon	xbinteg	xc_con	nmb_errs
ncutestall			

Other utilities should cause no harm.

C3800 Troubleshooting Guide

CHAPTER IV
(SWIP/SWIS)

1.* swis Failure Indications

The indications of a swis failure can be very misleading. When the swis loses contact with the key switch on the 3800, all screens will hang along with the mouse cursor.

When attempting to reboot the ConvexOS and xsfp screens must get a status from the key, through the swis. If this does not occur then the ConvexOS and xsfp screens can not be raised. This will force the other spu and console screens to fail, as well and abort the boot to the OPUS single user mode.

To verify that the swis is the problem it is necessary to edit the file /users/diaguser/.xinitrc (/.xinitrc). In this file place a "sleep 1000" entry as the next to last line in the file and reboot. This will allow the SPU window to stay up while troubleshooting is accomplished. The last line in the file will be "/diag/bin/xsfp -geometry +0+1000 > /dev/console 2>&1. This is the entry to bring the console window up. The sleep entry will be placed just prior to this entry.

If the swis is defective then a "dump_swis" command will generate errors. In addition any commands issued under pwr_util will hang.

2.* Isolating swip and NCU failures

It can be very difficult distinguishing swip problems from basic NCU failures with out the right knowledge. This task can be made much easier with the following information.

If the diaginit cannot be completed, there are no tools available to help in the diagnosis of a failure in the Swip/NCU loop. In order to clarify some points it is necessary to understand that there are accessible registers on the NCU. Particularly, there are loopback registers that are accessible for the swip. These registers are 210 and 214. The register 210 is actually the loopback register and 214 is the NCU header register.

If there seems to be nothing going on from the spu then it is possible to write and read these registers by the following method.

- 1) get 0x210
0x210 ff68c210 maybe returned (This will vary)
- 2) put 0x210 0x12345678
- 3) get 0x210
210 0x12345678 (return)

C3800 Troubleshooting Guide

If the proper value is not returned from these registers then it is very possible that the NCU has failed. This will generally happen if the NCU is not powered up. If the value only partially changes then there maybe a swip or swip cable failure.

It is important to know that the swip registers must be cleaned up before this can be accomplished. This was discussed in tech bulletin v3w5. If this is not accomplished then the results may lead to invalid conclusions.

3.* swip/swis Register Bit Definitions

The following is a list of bit definitions for all of the swip/swis registers. These individual outputs can be obtained by doing a get at dshell on the Opus of the individual register.

```
swip_bsrc_msb: 16#00
swip_bsrc_lsb: 16#10
swip_force_par_err: 16#00
swip_misc: 16#00
swip_int_level_ctl: 16#90
swip_int_ena: 16#80
swip_int_stat: 16#00
swip_force_int: 16#00
```

```
swip_bsrc_msb
VALUE: 16#00
| 15 | 14 | 13 | 12 # 11 | 10 | 9 | 8 # 7 | 6 | 5 | 4 # 3 | 2 | 1 | 0
|                                     # Inv| Int|Re- |NCU #   PERR
|                                     #Addr|Lock|run |Err #
```

```
swip_bsrc_lsb
VALUE: 16#10
| 15 | 14 | 13 | 12 # 11 | 10 | 9 | 8 # 7 | 6 | 5 | 4 # 3 | 2 | 1 | 0
|                                     # Inv| Inv| Inv| Err
|                                     #Addr|Size|Req | Ena
```

```
swip_force_par_err
VALUE: 16#00
| 15 | 14 | 13 | 12 # 11 | 10 | 9 | 8 # 7 | 6 | 5 | 4 # 3 | 2 | 1 | 0
|                                     # ILck|SOFT| | #
|                                     #Test| LOG|ADDR|DATA#   FORCE
```

```
swip_misc
VALUE: 16#00
| 15 | 14 | 13 | 12 # 11 | 10 | 9 | 8 # 7 | 6 | 5 | 4 # 3 | 2 | 1 | 0
|                                     # NXP|ADDR| NCU|RRun#
|                                     # ENA|ONLY| RES|Ena #
```

C3800 Troubleshooting Guide

```
swip_int_level_ctl
VALUE: 16#90
| 15 | 14 | 13 | 12 # 11 | 10 | 9 | 8 # 7 | 6 | 5 | 4 # 3 | 2 | 1 | 0
|                                     #   |   |   | #UART|UART|UART|UAR
|                                     # NCU| EXT|MODM| SPU# 3 | 2 | 1 | 0
|                                     unused
```

```
swip_int_ena
VALUE: 16#80
| 15 | 14 | 13 | 12 # 11 | 10 | 9 | 8 # 7 | 6 | 5 | 4 # 3 | 2 | 1 | 0
|                                     #   |   |   | Reserved
|                                     # NCU| EXT|
|                                     unused
```

```
swip_int_stat
VALUE: 16#00
| 15 | 14 | 13 | 12 # 11 | 10 | 9 | 8 # 7 | 6 | 5 | 4 # 3 | 2 | 1 | 0
|                                     #   |   |   | Reserved
|                                     # NCU| EXT|
|                                     unused
```

```
swip_force_int
VALUE: 16#00
| 15 | 14 | 13 | 12 # 11 | 10 | 9 | 8 # 7 | 6 | 5 | 4 # 3 | 2 | 1 | 0
|                                     #   |   |   | Reserved
|                                     # NCU| EXT|
|                                     unused
```

```
swis_bsrc_msb: 16#00
swis_force_par_err: 16#00
swis_misc: 16#0a
swis_int_level_ctl: 16#90
swis_int_ena: 16#1f
swis_int_stat: 16#00
swis_force_int: 16#00
key_switch: 16#17
```

```
swis_bsrc_msb
VALUE: 16#00
| 15 | 14 | 13 | 12 # 11 | 10 | 9 | 8 # 7 | 6 | 5 | 4 # 3 | 2 | 1 | 0
|                                     # Inv| Inv| Inv|
|                                     #Addr|Size|Req |
|                                     unused
```

```
swis_force_par_err
VALUE: 16#00
| 15 | 14 | 13 | 12 # 11 | 10 | 9 | 8 # 7 | 6 | 5 | 4 # 3 | 2 | 1 | 0
|                                     #   |SOFT| | #
|                                     # RES| LOG|   RES
```

```
swis_misc
VALUE: 16#0a
| 15 | 14 | 13 | 12 # 11 | 10 | 9 | 8 # 7 | 6 | 5 | 4 # 3 | 2 | 1 | 0
|                                     #   |Uart| #
|                                     # RES| RES|Rst |BPC4#BPC3|BPC2|BPC1|BPO
```

C3800 Troubleshooting Guide

```
swis_int_level_ctl
VALUE: 16#90
| 15 | 14 | 13 | 12 # 11 | 10 | 9 | 8 # 7 | 6 | 5 | 4 # 3 | 2 | 1 | 0
|           unused          # Mstr|      Interrupt Level
|           # Ena| 07 | 06 | 05 # 04 | 03 | 02 | 01

swis_int_ena
VALUE: 16#1f
| 15 | 14 | 13 | 12 # 11 | 10 | 9 | 8 # 7 | 6 | 5 | 4 # 3 | 2 | 1 | 0
|           unused          #      |      | #UART|UART|UART|UAR
|           #      RES      | Key# 3 | 2 | 1 | 0

swis_int_stat
VALUE: 16#00
| 15 | 14 | 13 | 12 # 11 | 10 | 9 | 8 # 7 | 6 | 5 | 4 # 3 | 2 | 1 | 0
|           unused          #      |      | #UART|UART|UART|UAR
|           #      RES      | Key# 3 | 2 | 1 | 0

swis_force_int
VALUE: 16#00
| 15 | 14 | 13 | 12 # 11 | 10 | 9 | 8 # 7 | 6 | 5 | 4 # 3 | 2 | 1 | 0
|           unused          #      |      | #UART|UART|UART|UAR
|           #      RES      | Key# 3 | 2 | 1 | 0

key_switch
VALUE: 16#17
| 15 | 14 | 13 | 12 # 11 | 10 | 9 | 8 # 7 | 6 | 5 | 4 # 3 | 2 | 1 | 0
|           unused          #      MODEM #      SPU
```

4.* SWIS Error Processing

There are two types of error detecting internal to the SWIS; hardware and software. Hardware errors are errors that require logging in the (1) BSRC. Software errors are detected through polling status registers internal to the OCTART.

Hardware errors are limited to problems in the SBus transaction. These include:

- * Invalid Address - The SWIS was given an address during an SBus transfer that was not mapped to any device or function.
- * Invalid Transfer Size - The SWIS received an SBus extended transfer request.
- * Invalid Transfer Request - The SWIS received an SBus burst transfer request.

Hardware errors result in the process making the access receiving a bus error signal (SIGBUS). If the process doesn't have a bus error handler, it will be terminated by the kernel.

C3800 Troubleshooting Guide

Software errors are determined through register polling of the OCTART. These include:

- * Framing Error - A stop bit was not detected when the data character in the OCTART FIFO was received.
- * Parity Error - A parity error was detected when the data character in the OCTART FIFO was received.
- * Overrun Error - A new character was received while the OCTART FIFO was full and a character was in the receive shift register.

Software errors are reported by the kernel in the console window. These errors are considered non-fatal, so the kernel attempts to sync back up to the BPC and continue operation.

- (1) BSRC (Bus Error Source Register) - Refer to the Chapter IV (SWIP/SWIS) of the C3 Maintenance Handbook for register bit definitions.

5.* SWIP Error Processing

There are two sources of errors from the SWIP: internal and NCU. Both of these sources perform error logging in the BSRL and BSRU registers.

Internal errors are limited to problems in the SBus transaction. These include:

- * Invalid Address - The SWIP was given an address during an Sbus transfer that was not mapped to any device or function.
- * Invalid Size - The SWIP received an SBus extended transfer request.
- * Invalid Request - The SWIP received an SBus transfer request.

These errors result in the process making the access receiving a bus error signal (SIGBUS). If the process doesn't have a bus error handler, it will be terminated by the kernel.

The NCU may generate several errors. These errors may be encountered when a transaction process is attempted or due to static error conditions related to the state of the NCU.

Anytime a NCU error condition results in a error acknowledgment being performed (SIGBUS), the SOFT_LOG bit will be set in the PEFR register. Soft Log performs two vital functions: 1) preventing the error processing code in the SPARC SPU from receiving another SIGBUS from some follow on SWIP access before it is ready to handle it and 2) disabling all accesses to the NCU, thus preserving transaction particulars that might prove useful in debugging the problem. The SWIP NCU interface state machine will not attempt a transaction while SOFT_LOG alone is set. The state machine may be re-enabled either by clearing SOFT_LOG, or by also setting CHECK_ERROR. When both CHECK_ERROR and SOFT_LOG are set the state machine will operate, but error acknowledgments will not occur. Each transaction performed under these conditions must be followed by polling of all error reporting bits in the BSRU and BSRL. Once all of the error recovery/correction processing has been performed, the software driver should clear SOFT_LOG, if it is appropriate to do so. This means all error codes have been accounted for and cleared.

- * Interlock - The SPU-NCU cable is not completing the interlock connections between the SPARC SPU and the C38XX. The interlock error will not cause an error acknowledgment unless the ERROR-ENABLE bit is set in the BSRL. The error will then be reported on the next access attempt to the NCU.
- * Rerun - The RERUN error bit is set in the BSRU any time the SWIP board must time out an access attempt to the NCU. A rerun error should always be followed by a toggling of the RERUN_ENABLE bit.
- * NCU ERROR - The NCU may issue an error to the SWIP under three different circumstances. Error assertion is the default state of the NCU ERROR signal whenever the NCU does not have power applied to the board; assertion registers on the SWIP provide this default state. The NCU_ERROR bit may only be cleared after the NCU power has been applied.

The NCU will also assert the NCU ERROR signal whenever it detects a parity or invalid address error in the address header that the SWIP has latched into the NCU. The problem is generally related to hardware transceiver failure on either the SWIP or NCU, or a cable connection problem. A write/read pattern test of the 2x214 Address Loopback Register will usually detect the failed signal.

NCU ERROR assertion is performed anytime data parity failure is detected on data sent to the NCU by the SWIP during a write transaction. A read of the NCU ERROR LOG register 0x401c will indicate if the data parity bit is set, along with associated parity syndrome bits for the corrupted bytes of the transaction. This error is normally related to failed hardware.

Operations internal to the NCU and NXP interface on the NCU may cause error conditions to be set in the NCU ERROR LOG. These registered error conditions are logically OR'd and the result sent to the SWIP via the NCU ERROR signal. The NCU ERROR will be asserted during a transaction if one of these conditions has been set. A transaction does have to be in progress for a condition to be set, but the SWIP will not asynchronously act on the assertion. A transaction to the NCU must be in progress before an error acknowledgment will be issued at the end of the transaction.

PERR - The Parity ERROR bits are the failed parity syndrome check bits for the 4 data bytes during a read from an NCU transaction. This problem is generally related to a hardware transceiver failure on either the SWIP, NCU, or a cable connection problem. A write/read pattern test of the 0x214 Address Loopback Register, will usually detect the failed signal. The PERR bits should never be found set at the same time as the NCU ERROR reporting bit in the BSRU. This condition is usually indicative of a NCU powerdown during a read transaction.

Note: For register bit definitions, see Chapter IV (SWIP/SWIS) in the C3 Maintenance Manual.

6.* Intermittent swis Errors

It has, very recently, been discovered that the swis is experiencing a very intermittent and so far, minor symptom.

It appears that when powering up the system with diaginit, that the swis will complain that a specific uart channel is missing. In most cases the uart channel will be for bay 6, which is a test loop on the swis itself and therefore of little consequence.

But, there is a possibility that the failure can occur with an active bay; particularly bay 4 and thus prevent this bay from coming up. If this should happen, it is possible to clear the failure by executing the UART related subtests (213 and 214) in spu4000.

At present, it is felt that the problem is timing related with the spu and a solution is being investigated. An example is shown below:

```
Sep 2 19:21:18 neptune_spu savecore: reboot after panic: parity error
Sep 2 19:21:46 neptune_spu vmunix: bpc: interlock found on uart channel(0)
Sep 2 19:21:46 neptune_spu vmunix: bpc:interlock found on uart channel(1)
Sep 2 19:21:46 neptune_spu vmunix: Swis:interlock changed interrupt received
and not caught
Sep 2 19:21:46 neptune_spu vmunix: bpc: interlock found on uart channel(2)
Sep 2 19:21:46 neptune_spu vmunix: bpc: interlock found on uart channel(3)
Sep 2 19:21:46 neptune_spu vmunix: swis: interlock changed interrupt
received and not caught
Sep 2 19:21:46 neptune_spu vmunix: bpc: interlock found on uart channel(4)
Sep 2 19:21:46 neptune_spu vmunix: swis: interlock changed interrupt
received and not caught
*****uart channel (6) missing*****
Sep 2 19:21:46 neptune_spu vmunix: bpc: interlock found on uart channel(7)
Sep 2 19:21:46 neptune_spu vmunix: swis: interlock changed interrupt
```

93/02/22
07:51:00

(trngps)(mikey:throg Job: vol.8 Date: Mon Feb 22 07:50:58 1993)

psfilter.in.23447

52

C3800 Troubleshooting Guide

CHAPTER V
(MISC)

1.* C3800 Transformer Requirement

CONVEX has decided that the transformer is absolutely necessary for all 3800 installations. Mechanical Engineering feels that this is necessary for the following reasons:

- 1) The FCC validation was received with a transformer in the configuration. So, it is necessary to maintain this validation.
- 2) For impedance matching between the AC source and the system.
- 3) For phase balancing within 5%.

Again, it should be explained to the customer that this transformer is a requirement for a C3800 installation.

2.* C3800 Board Damage During Shipping

We have, recently, experienced some minor damage to C3800 boards being returned. The damage has been minimal, but usually involves some minor buckling of the frame. We believe that the cause of this is elevating the power pallet to the top.

To avoid this, stickers will be applied to the outside of the box indicating "This Side Up". In addition it is asked that board be placed in the box so that the power pallet is always at the end of the box where the wheels would mount. This is easy to determine as there is a small cutout at the opposite end where the silver handle on the board is meant to rest.

This cutout is larger than the handle in order to accommodate left and right hand boards. Please make certain that the handle fits into this cutout. This may involve turning the board over.

It is also requested that the boards be returned, packaged exactly as they are received on site.

Following these few suggestions will reduce possible damage to the boards and aid in turn around time to the field.

3.* C3800 Crossbar Power Pallet Nutplates

It should be understood that the nutplates (p/n 320-002189-500) should not be included when returning C3800 crossbar power pallets. The reason for this is that the nutplates have been removed from the bom of the power pallet (p/n 500-0005000-200).

If the nutplate is returned then there will not be one available for the replacement. This would generally not be a problem as the current power pallet would generally not be returned until receiving the replacement, but could be a problem if the bad part is returned first.

C3800 Troubleshooting Guide

4.* Multiple Boot Scripts

It should be understood that there are two boot scripts for the C3800. The primary script is located in /mnt/os, on the SPU, and should be executed when in the CONVEX_OS screen under rmtdiag. The other script is located in /diag/bin.

The second boot script in /diag/bin is really a redirect from the spu window to the CONVEX_OS window. This one should be executed normally. The script appears below:

```
#!/bin/sh

if /diag/bin/sfp -k OFF; then
echo "Key is in OFF position";
else
echo 'Boot command sent to CONVEXOS CONSOLE window. Look there for output.'
echo "cd /mnt/os; /mnt/os/boot $1" > /tmp/console_command;
chmod 666 /tmp/console_command;
kill_by_name CONVEXOS_CONSOLE -USR2;
fi
```

Executing the boot command from /mnt/os in the spu window means that this redirect to the OS window will not take place and so the boot will not function as planned.

It is, therefore, recommended that no boot be initiated under /mnt/os, or that the full path for the boot be used. In other words, initiate the boot process with /diag/bin/boot. This will avoid any unnecessary problems.

5.* Kernel Damage During a Virgin Install

The C3800 has proven vulnerable to kernel damage occurring during a virgin install of the OS. Even though the problems have only been associated with a virgin install, so far, this problem should be considered after any sysgen that is accomplished on the system.

This appears to be a 3800 problem, only, and has not been duplicated on any other system type.

The symptoms are that the fortran compiler will intermittently abort with the following error:

```
Compiler abort/assert may have been caused by source on 159.7.
fc: /usr/convex/fskel was terminated by signal 'SIGILL'
```

By intermittent, I mean that the compile will fail 60-70% of the time. This failure will occur on all source files and will always report on the same line for each source. The above example indicates that the failure occurred on line 159. The error is occurring with fskel and is an illegal instruction.

Other symptoms associated with this problem are as follow:

C3800 Troubleshooting Guide

- 1) Disabling the dcache will cause the situation to improve drastically
- 2) The problem will improve as the physical memory size is reduced
- 3) There is no head configuration that will affect the situation

It appears that the problem is generated during a sysgen and involves damage to the kernel itself.

The solution to the problem is somewhat painful and involves, first, attempting to move the effected file system (both root and usr) to another disk. This can be accomplished by a cpall, or a dump piped to the restore command.

If this fails to resolve the problem, then it will be necessary to perform a virgin install of the OS.

6.* Criteria for Board Replacement

The following guidelines are currently being used to determine replacement of boards in the field. They are listed here to familiarize the field with these guidelines and is not intended for the field to make their own determinations in the matter of replacement.

- 1) On the first occurrence of a STRAM. These component failures are generally indications of a failed part and so will be considered for immediate replacement.
- 2) On the occupance of 2 successive purge ram failures, indicating the same ram and the same address within the ram.
- 3) Three, apparently, unrelated failures on the same NSP.
- 4) The occurrence a nvrfl2 on either NSP, or NVP, after the boards have been separated. A board will not be replaced until there is evidence that it has followed a moved board. The part replaced, in this instance, will generally be a NVP.
- 5) Most other failures will be considered a hard failure and result in replacement as warranted.

7.* Replacing the Crossbar Backplane

The following procedure can be followed when replacing a crossbar backplane in the field:

- 1) Remove the crossbar boards (7) and place in static free containers.
- 2) Remove rear crossbar door.
- 3) Remove all ribbon cables and other power and sensor connectors from backplane.
- 4) Remove the card cage. This is attached to the backplane by 13 screws, 5 front and 8 rear.
- 5) Remove rear xbar airflow sensor.

C3800 Troubleshooting Guide

- 6) Remove even and odd side xbar power pallets. These are fastened by 2 screws in the rear and one forward.
- 7) Remove xbar clock cables on the underside of the backplane.
- 8) Fashion an epont restraining tool. This can be accomplished with twine, or a coat hangar. The purpose is to force the epont connectors to rest just above the backplane.
- 9) Remove all epont connectors and suspend above backplane with tool.
- 10) Place bags, or dust covers over eponts.
- 11) Remove 6 screws holding backplane to cabinet. The two rear screws have no nuts, but the 4 forward screws are held with nuts.
- 12) Remove backplane by lifting slightly and sliding rearward.
- 13) Move power pallet mounting plates to new backplane.
- 14) Clean epont/fuji poly before installing new backplane.
- 15) Slide new backplane in from rear.
- 16) fasten backplane in place with 6 mounting screws.
- 17) Re-install epont connectors to backplane, carefully removing restraint tool.
- 18) Reinstall card cage.
- 19) Install both power pallets.
- 20) Install air flow sensor.
- 21) Install clock cables.
- 22) Re-install all removed cables from backplane.
- 23) Re-insert xbar boards.
- 24) Install rear door.

8.* ConvexOs 10.2 patches

We currently have a patch that addresses the return of the nvrfl failures with 10.2 OS.

The patch is 10.2.141. This patch will only be used with the current Beta release of 10.2, as the patch will be included in the production release.

At current, the production release of 10.2 is not scheduled to finish qual before the end of October. This schedule will only be met if no major problems are discovered.

At this time, the beat release of 10.2 with all available patches, is equivalent to the production release.

The current patches available, for 10.2 are as follow:

- | | |
|----------|---------------------------------|
| 10.2.128 | SCSI driver |
| 10.2.129 | VME Async printer |
| 10.2.131 | crashes caused by stripe driver |
| 10.2.132 | Tape driver |
| 10.2.133 | Stripe hangs |
| 10.2.134 | crashdump |
| 10.2.135 | Customer Specific Patch |
| 10.2.137 | Ethernet Driver |
| 10.2.138 | PTE Violations |
| 10.2.139 | Tape System |
| 10.2.140 | Stripe driver related hangs |
| 10.2.141 | NVRFL Patch |
| 10.2.142 | New cpu_monitor for APR |

Of course, 10.2 OS for specifically for 3800's and should not be used on any other system.

C3800 Troubleshooting Guide

CHAPTER VI
(POWER)

1.* Power Brick Locations

The following will indicate locations of the power bricks located on the power pallets of C3800 boards. All locations are slightly skewed for each board, so each board will be displayed separately.

NOTE When replacing a power brick, it is recommended that the accompanying fuse be replaced and vice versa.

All displays from heat sink side left facing as viewed from heatsink side.

N C U P P

SPARE	SPARE			
VEE -4.5	VTT -2	VEE -4.5	VEE -4.5	VEE -4.5
F S6	E S7	E S8	D S9	C S10
VTT -2	VTT -2	VTT -2	VEE -4.5	VEE_IOK-5.2
B S1	C S2	D S3	B S4	A M5
VTT_GA -2	VTT -2	VCC +5	VEE -4.5	
A M1	A M2	A M3	A M4	

C3800 Troubleshooting Guide

N V P P P

VTT_GA -2	VTT -2	VEE -4.5	VTT -2	VTT_GA -2
H S6	C S7	B S8	D S9	C S10
VTT_GA -2	VTT_GA -2	VTT -2	VTT_GA -2	VTT_GA -2
B S1	C S2	B S3	E S4	F S5
VTT_GA -2	VTT -2	VTT_GA -2	VEE -4.5	
A M1	A M2	D S13	A M4	

C3800 Troubleshooting Guide

N S P P P

VTT_GA -2	VTT -2	VEE_IOK-5.2	VTT -2	VTT_GA -2
F S10	D S9	B S8	C S7	G S6
VEE_IOK-5.2	VTT_GA -2	VTT -2	VTT_GA -2	VTT_GA -2
A M5	E S4	B S3	C S2	B S1
	VEE -4.5	VTT_GA -2	VTT -2	VTT_GA -2
A M4	D S13	A M2	A M1	

C3800 Troubleshooting Guide

N M B P P

VEE -4.5	VTT -2	VCC +5	VTT_GA -2	VCC +5
D S10	E S9	B S8	B S7	C S6
VEE -4.5	VEE -4.5	VTT -2	VTT -2	VTT -2
C S5	B S4	D S3	C S2	B S1
	VEE -4.5	VCC +5	VTT -2	VTT_GA -2
A M4	A M3	A M2	A M1	

C3800 Troubleshooting Guide

C C U P P

VCC +5	VCC +5	VCC +5	VCC +5	VCC +5
D S10	E S9	F S8	G S7	H S6
VEE -4.5	VEE -4.5	VEE -4.5	VCC +5	VCC +5
B S5	C S4	D S3	B S2	C S1
	VEE -4.5	VCC +5	VTT -2	-5VDC -5
A M4	A M3	A M2	A M1	

C3800 Troubleshooting Guide

N I A P P

VTT -2	VEE -4.5	VEE -4.5	VTT -2	VEE -4.5
D S10	C S9	D S8	E S7	E S6
VEE_IOK-5.2	VEE -4.5	VTT -2	VTT -2	VTT_GA -2
A M5	B S4	C S3	B S2	B S1
	VEE -4.5	VCC +5	VTT -2	VTT_GA -2
A M4	A M3	A M2	A M1	

2.* C3800 PPC Brick Fuses

There are two fuses associated with each power brick on each ppc. These are 1/2 amp and 3 amp fuses. The resistance of each fuse, cold out of the circuit, measures:

1/2 amp = 1.65 ohm 3 amp = .071 ohm

93/02/22
07:51:00

(trngps)(mikey:throg Job: vol.8 Date: Mon Feb 22 07:50:58 1993)

psfilter.in.23447

57

C3800 Troubleshooting Guide

3.* C3800 Crossbar Power Pallets

Be aware that there are two crossbar power pallets in the C3800. There is an even and odd side power pallet. The even side power pallet supplies power to the XCL as well as all of the even side xbar boards. In addition the power pallet controller for both power pallets, is located on the even side pallet.

In order to reduce costs, only the odd side pallet is provided as a spare. There is no problem here, as it is relatively painless to convert the odd side pallet to the even side.

To accomplish this, it is necessary to remove the mounting hardware, remove the plugs on the mounting holes on the supply and mount the new supply to the even side bracket.

It should be understood that the even side supply is under greater load, so in emergencies, it may be possible to move a weak pallet from the even side to the odd side until a replacement can be secured.

In order to verify that the even pallet may work on the odd side, it is possible to remove the XCL and power the xbar up using pwr_util. If the crossbar will power up with the XCL removed, it is a candidate to be moved to the odd side. The xbar can be powered up manually by using the k and l options of pwr_util.

If the odd pallet fails then there is little that can be done except to replace it.

4.* C3800 Power Problem Isolation

It is important to understand that power problems can play a part in many hard crashes and not necessarily be very obvious. In many cases the component that reports the crash is not the one experiencing the power failure.

Quite often the failure will be lost in the noise, but upon closer examination a power problem will be found, either by examining the "display_log", or by examining the error more closely. When this happens the error will be reported, quite often, as a PPC failure. This PPC failure will be followed by a number, as

#BPC: 0 PPC: 4 Warning: 300 volts removed from pallet
#BPC config error code: 00 PWR config error code: 19

C3800 Troubleshooting Guide

This failure indicates a problem with the power pallet on board number 4 in bay 0. To determine the correct location of the failure it is necessary to count across bay 0 to the correct board. It is important to understand the correct sequence and the fact that slots 0 and 7 will, likely, be empty. In this case the failure will be with NVP 1.

0	1	2	3	4	5	6	7
C	N	N	N	N	N	N	C
C	M	S	V	V	S	M	C
U	B	P	P	P	P	B	U
P							P
P							P

In addition, the BPC status display at the time of boot, or from pwr_util can be used to isolate the failed module. In the case below, again the failure is identified as slot 4. Notice that slots 0 and 7 indicate no activity.

```
[K#--> Bay Power Update Returned 00 ppc mask for bpc uart 0
[K#SW Info (DiagIN236): Bay Power Controller status message
[K#
[K#PPC Status for BPC number 00
[K# bay pp fw prt bkpln bkpln plt bd bd bd bd bd bd bd bd bay bps bus bus
[K# cnf cnf rev id slot type typ 0 1 2 3 4 5 6 7 pwr num OK on
[K#0|ff ff ffff ff ff ff ff ff ff ff ff ff ff ff f3 ff ff ff
[K#1|00 00 0205 00 06 02 01 01 00 00 00 00 00 00 00 00 00 00 00 00
[K#2|00 00 0205 00 07 02 02 02 00 00 00 00 00 00 00 00 00 01 00 00
[K#3|00 00 0205 00 08 02 06 06 00 00 00 00 00 00 00 00 00 01 00 00
[K#4|00 00 0205 01 09 03 06 06 ff ff ff ff ff ff ff 19 02 01 ff
[K#5|00 00 0205 01 0a 03 02 02 00 00 00 00 00 00 00 00 00 04 00 00
[K#6|00 00 0205 01 0b 03 01 01 00 00 00 00 00 00 00 00 00 05 00 00
[K#7|ff ff ffff ff ff ff ff ff ff ff ff ff ff ff f3 ff ff ff
```

It is, also, quite common for the board reporting a hard error to have nothing at all to do with the failure. So hard errors should be examined very closely to determine their actual meaning.

The process of troubleshooting a power problem can be an easy one if the proper procedures are followed and the tools are used to the fullest advantage. This is intended to be a guide to aid in the isolation of the source of power failures.

Below is the output from pwr_util, using the "n" option for display of the status of an individual power pallet. As can be seen below, ppc #2 has been selected, which as explained in the "C3800 Troubleshooting Hints" Chapter 6 Article 4, is NSP0. It is recommend that this article be read at this time to get the total benefit of this power discussion.

The utility pwr_util can be complemented with powermon, but this status display will contain everything that is necessary.

93/02/22
07:51:00

(trngps)(mikey:throg Job: vol.8 ate: Mon Feb 22 07:50:58 1993)

psfilter.in.23447

58

C3800 Troubleshooting Guide

spu> pwr_util

- a) Bpc chk interlock
- e) Offline bpc
- i) Cop Read
- m) Power Down
- r) Busses off
- q) Quit
- b) Reset uart
- f) F/W revision
- j) Cop Write
- n) Send Status
- t) Transparent
- ?) Print Menu
- c) Bpc reset
- g) F/W Download
- k) Bay Power chk
- o) Set voltage
- d) Init bpc
- h) Bay Config chk
- l) Power On
- p) Set temp

Enter command: n
 Select uart channel [0-4]: 0
 Select ppc id [0-8]
 where 8 indicates bay controller only: 2

+++>
 <Tue Jun 30 08:02:55 1992> pwr_util:../pwr_util.c:230
 SW Info (DiagIN238): Power Pallet Controller status message

PPC Status
 fill fw pp open err err plt bd bd bd bd bd bd bd bd slot warm hot bus
 byte state fail ilck grp code id 0 1 2 3 4 5 6 7 id flag flg trim
 00 02 00 00 f1 1f 02 02 00 00 00 00 00 00 00 07 00 00 00

bus_on brick brick brick brick tmp0 tmp1 tmp2 tmp3 tmp4 tmp5 tmp6 tmp7 A/D
 error exist type fuses input snsr snsr snsr snsr snsr snsr snsr over
 00 3fff 3bdb 0000 0000 57 58 58 56 54 5b 5e 5a 00

hous A/D0 A/D1 A/D2 A/D3 A/D4 A/D5 A/D6 A/D7
 volt cnvt cnvt cnvt cnvt cnvt cnvt cnvt cnvt
 00 f7fe 0813 ffff f7ad fcb8 fce2 fffe f8ca

Below is a list of definitions for entries in the example above:

- 1) brick exist - A "1" is set for every brick installed on the pallet. A different bit is used to represent each brick installed. The below list is used for all brick and fuse status. Where S10 is slave supply 10 and M1 is master supply 1.
 S10=0001 S9=0002 S8=0004 S7=0008 S6=0010 S5=0020 S4=0040 S3=0080
 S2=0100 S1=0200 M4=0400 M3=0800 M2=1000 M1=2000
 example: 3fff indicates that all supplies are installed.
- 2) brick input - Is the opposite of brick exists, as a "1" indicates that the individual supply is turned off. This is a great aid in determining which power busses are defective.
- 3) brick fuses - A "1" indicates which fuses are open. In addition to the list of supplies above, this entry has two additional bits.
 FUSED300V=4000 INPUT300V=8000

C3800 Troubleshooting Guide

4) brick type - This entry will indicate the type of bricks installed, where a "1" indicates a 2V brick and a "0" indicates a 5V brick.

The A/Dx fields are a 2's complement hex equivavlent representation of a DC voltage level. To convert to a DC voltage the 2's complement hex value converted to decimal and multiplied by 5/2048. It is important to understand that values of the 0xxx is positive and fxxx is negative.

Example: A/D1 = 0813 converts to 2067 decimal * 5/2048 =5VDC
 A/D0 = f7fe converts to 0802 hex and 2050 decimal * 5/2048=-5VDC

Those values that are a 0000 would generally indicate a shorted input. The master supply would be suspect here. A value of ffff would mean that this bus does not exist on this board.

The table below indicates the bus covered by each A/D circuit on a board by board basis.

BOARD	A/D0	A/D1	A/D2	A/D3	A/D4	A/D5	A/D6	A/D7
NMB	PPCM5V	PPCVCC	GND	GND	VTGA	VTT	VCC	VEE
NSP	PPCM5V	PPCVCC	GND	VEE10K	VTGA	VTT	GND	VEE
NVP	PPCM5V	PPCVCC	GND	GND	VTGA	VTT	0	VEE
NIA	PPCM5V	PPCVCC	GND	VEE10K	VTGA	VTT	VCC	VEE
NCU	PPCM5V	PPCVCC	GND	VEE10K	VTGA	VTT	VCC	VEE
XBAR	PPCM5V	PPCVCC	VEE	VEE	VTGA	VTGA	VTT	VTT
CCU	PPCM5V	PPCVCC	GND	GND	M5V	VTT	VCC	VEE

Where: M5V = -5VDC
 VCC = 5VDC
 VTT = -2VDC
 VEE = -4.5VDC
 GND = 0VDC
 VEE10K= -5.2VDC (STRAM's)
 VTTGA = -2VDC

The temperature sensors located on the boards (tmp0-7) indicate the fahrenheit temperatue in 10mv increments. The conversion is as follows:

adc counts * (2.5/.01*255)
 2.5/.01*255 = .98
 adc count *.98
 convert hex value of adc count to decimal and multiply by .98.

example: tmp0=57 0x57 converts to 87 dec.
 87*.98 = 85.26 degrees F

The other fields can be identified by use of the C38XX Power System Firmware Description.

With this document and the Firmware Description, it should be relatively easy to identify and correct Power Pallet related failures.

C3800 Troubleshooting Guide

5.* C3800 Voltage Margins

It is possible to margin voltages on a C3800 in the following format:

```
altsetpts -vee10k -5.304 sp0 spl
altsetpts -vee -4.590 sp0 spl
altsetpts -vtt -2.04 sp0 spl
altsetpts -vttga -2.04 sp0 vp0
```

The utility "altsetpts" is only temporary and when the system is rebooted, the voltages will return to their preset values. However, "altsetpts" is the only means to margin voltages on individual boards.

The permanent means of margining voltages is as follows:

- 1) edit the file /diag/db/lab_machine from a 0 to a 1

```
file appears as "/diag/bin/cdb_update lab_machine 0"
^
change to 1
```

- 2) Edit file /diag/db/set_lab_busses, changing voltages to desired values. There are several categories of voltages in this file and care should be taken to avoid changing any other except those voltages listed under lab values.

This file will appear as follows:

```
#set nominal values for lab machines (When lab_machine 1)
#lab values
vcc="5"
vee="-4.5"
vtt="-2"
vttia="-1.905"
vttsp="-1.905"
vttvp="-1.905"
vttcu="-1.905"

vttga="-2"
vttgaia="-2.07"
vttgasp="-2.07"
vttgavp="-2.07"
vttgacu="-2.095"

vee10k="-5.2"
m5v="-5"
xvee="-4.5"
xvtt="-1.905"
xvttga="-2.1"
ppcvcc="5"
ppcm5v="-5"
```

C3800 Troubleshooting Guide

```
#production values (These are used when lab_machine is set to 0)
vcc_p="5"
vee_p="-4.5"
vtt_p="-2"
vttga_p="-2"
vee10k_p="-5.2"
m5v_p="-5"
xvee_p="-4.5"
xvtt_p="-2"
xvttga_p="-2"
ppcvcc_p="5"
ppcm5v_p="-5"
```

- NOTE* It should be understood that no margins should be changed in this file unless instructed to do so. It should also be understood that when changing one voltage value, all others must be moved to nominal, or problems will result.

6.* EPROM Damage When Powering Down A NIA Board

Manufacturing has reported that on some occasions after powering down the NIA board, the eprom gets blown on the ccus (primarily the IDCs) in the right ccu backplane when powering the NIA back up (The left ccu backplane is powered by the same bps as the NIA but the right backplane is powered by a separate bps.)

Based on these observations and until further analysis can be performed, it is recommended that the NIA not be powered down without also powering down the right ccup for those C3800 systems having ccu boards installed in the right ccu backplane.

7.* C3800 Power Transformer Specification

It has been decided that CONVEX will no longer insist on the CONVEX supplied transformer's use with C3800 installations. This more lenient policy is spelled out below and originates from the CONVEX mechanical engineering organization.

As usual, if there are any questions, please contact Hardware Support.

C3800 AC Power Source Specification

This is a brief discussion of the use of alternate power source equipment by Convex C3800 customers. This is not intended to replace the existing site preparation guide, but rather to supplement the guide until this new information can be incorporated into it.

93/02/22
07:51:00

(trngps)(mikey:throg Job: vol.8 -ate: Mon Feb 22 07:50:58 1993)

psfilter.in.23447

60

C3800 Troubleshooting Guide

The C3800 power system was designed utilizing a Convex specified power isolation transformer as a front end. The low pass filter characteristics and balanced impedance of the Convex isolation transformer were designed in as a part of the CPU power system to give superior noise immunity and phase balance performance. The system was developed and tested around this specific equipment which can be provided to each customer by Convex to help ensure performance comparable to machines located at the Convex facility.

Convex's position on the use of alternate power sources is now open as long as certain requirements are met. This was not originally Convex's position since all the original emission and immunity tests were performed at the primary of the Convex transformer, requiring its usage. Since that time, more testing has occurred confirming that most any high quality transformer or PDU can perform the required isolation and attenuation functions.

Convex installation documentation is very specific about the C3800 CPU complex being power isolated from all other equipment with an isolation transformer. This isolation not only protects the machine from noise and surges but other site equipment from any surges generated by the Convex equipment. Even Convex peripherals are not allowed on the same transformer secondary.

The use of alternate transformer equipment should meet the following general criteria. The requirement for isolated power dedicated to the C3800 CPU complex does not change with alternate equipment usage.

PERFORMANCE CHARACTERISTICS OF EXISTING OR NEWLY PURCHASED PDU EQUIPMENT -

The PDU output must be fully isolated and dedicated to C3800 CPU complex only. (Connection of Convex SPU workstation allowed)

The PDU must be at least 50KVA capacity with a minimum K factor of 7 to handle the fundamental and harmonic currents.

The PDU must meet all local codes of the install site for performance and safety.

The PDU must provide a separate 60 Amp circuit to each of up to five C3800 computer bays. (Present connections require 5100C9W or 5100R9W receptacles available from Hubble, Mennekes, or Convex. Future machines will switch to a SP560C9 or SP560R9 connector. Customers will be informed of such a change in advance of site preparation. 5100 plugged equipment will be available on future expansion equipment delivered to sites already prepared with such connectors)

The PDU must be located within 100 feet of the Convex C3800. Outlets from the PDU must be within 10 feet of the Convex C3800.

The maximum main breaker on the input to the PDU must not exceed 125% of the PDU tagged rating.

The isolation transformer within the PDU shall provide full isolation and have a maximum effective coupling capacitance between primary and secondary of 33 picofarads.

Attenuation of line noise and transients shall equal or exceed the following limits at full load and unity power factor:

C3800 Troubleshooting Guide

A) COMMON MODE
0 to 1.5KHZ - 120DB; 1.5KHZ to 10KHZ - 115DB; 10KHZ to 100KHZ - 110DB;
100KHZ to 1MHZ - 60DB.

B) TRANSVERSE MODE
1.5KHZ to 10KHZ - 60DB; 10KHZ to 100KHZ - 62DB; 100KHZ to 1MHZ - 56DB.

The PDU is to provide surge protection such that output surge levels do not exceed IEEE 587 category A levels.

8.* Site Prep Guide Power

The "C3800 Site Preparation Guide" lists power requirement for individual components in table 1-14 thru 1-17. These values were theoretical in nature, when released. Below is listed the actual measured values for these components.

COMPONENT	WATTS
Central Cabinet	1200
I/O Bay Cabinet	1200
NIA	1050
NCU	950
CCU	200
CCU BPS	100
CPU Bay Cabinet	1200
NMB	1200
CPU (VP and SP)	2000

These values are all maximums and individual configuration values do not exist. For example, there is no information available on power dissipation of NMB's of different sizes. So, as these figures are maximum, it is recommended that these figures be used for all configurations.

9.* PPC and BPC Failures - EEPROM Not Write Protected

It has been discovered that PPC's and BPC's are vulnerable to catastrophic failure. The reason is because the EEPROM's on these devices are not write protected and so can be written to accidentally. Although this is a real possibility, it has only been witnessed 3 times in a field environment.

The symptoms exhibited, when this happens, is that an individual board, or single bay will power down during operation, for no apparent reason. It will then be impossible to power this particular failed component up again.

*****NOTE***** Boards that power down, but come right back up are not included in this situation.

93/02/22
07:51:00

(trngps)(mikey:throg Job: vol.8 - ate: Mon Feb 22 07:50:58 1993)

psfilter.in.23447

61

C3800 Troubleshooting Guide

If this particular situation occurs on a system, it is recommended that the firmware be downloaded by pwr_util. If it is the write protect failure, then this will clear the problem. Replacing the PPC, or BPC will also correct the problem, but download of firmware is far quicker and cheaper.

If this should occur, please contact the TAC.

This problem is being address and a solution, a new loader, will be released in 3.1 of the diagnostics. This loader will allow the write protection of these EEPROM's.

10.* Board Temperature Sensors

The following is a picture of a powermon on the I/O bay and CPU bay. Below each picture is a listing identifying the temperature sensors in each bay.

I/O Bay Sensors

```

----- PPC Status -----
config f/w pt backplane plt brd brd brd brd brd brd brd brd bay bps bus bus
by pp rev id slot type typ -0- -1- -2- -3- -4- -5- -6- -7- pwr num OK on
0| ** ***** **
1| ** ***** **
2|00 00 2.8 08 07 IO_B IA IA 00 01 00 00
3|00 00 2.8 08 08 IO_B CU CU 00 00 00 00
4| ** ***** **
5| ** ***** **
6|00 00 2.8 09 06 XBAR XPB XPB XRT XS1 XS0 XCL XS0 XS1 XRT 00 00 00 00
7|00 00 2.8 08 0c CCUR CCU ERR 00 01 00 00

```

```

----- Temperature Status -----
--- Status --- --- Outlet (C) --- --- Inlet (C) ---
f/w rev Sensor BdErr warm hot warm hot
2.16 00 00 49 63 49 63
Temp Sensors: 1 2 3 4 5 6 7 8
Status: OK OK OK OK OK OK OK OK
Temp (C): 30 10 10 10 25 25 10 10
Fan Sensors: 0 1 2
Status: OK OK OK
Value: 00 00 00

```

Temp sensor 1 - in xbar outlet near central cabinet.
Temp sensor 2 - Left ccu sensor.
Temp sensor 3 - Bay4 outlet - in blower cage.
Temp sensor 4 - Right ccu sensor.
Temp sensor 5 - in xbar inlet in front of door fans.
Temp sensor 6 - Intake (directly below and in front of boards).
Temp sensor 7 - Not used.
Temp sensor 8 - Not used.
Reading of 10 on temp sensor - sensor most likely disconnected.

C3800 Troubleshooting Guide

CPU Bay Sensors

```

----- PPC Status -----
config f/w pt backplane plt brd brd brd brd brd brd brd brd bay bps bus bus
by pp rev id slot type typ -0- -1- -2- -3- -4- -5- -6- -7- pwr num OK on
0| ** ***** **
1|00 00 2.8 00 06 CPUL MB MB 00 00 00 00
2|00 00 2.8 00 07 CPUL SP SP 00 01 00 00
3|00 00 2.8 00 08 CPUL VP VP 00 01 00 00
4|00 00 2.8 01 09 CPUR VP VP 00 04 00 00
5|00 00 2.8 01 0a CPUR SP SP 00 04 00 00
6|00 00 2.8 01 0b CPUR MB MB 00 05 00 00
7| ** ***** **

```

```

----- Temperature Status -----
--- Status --- --- Outlet (C) --- --- Inlet (C) ---
f/w rev Sensor BdErr warm hot warm hot
3.16 00 00 49 63 49 63
Temp Sensors: 1 2 3 4 5 6 7 8
Status: OK OK OK OK OK OK OK OK
Temp (C): 10 10 17 10 10 14 10 10
Fan Sensors: 0 1 2
Status: OK OK OK
Value: 00 00 00

```

Temp sensor 1 - Not used.
Temp sensor 2 - Not used.
Temp sensor 3 - Bay outlet - in blower cage.
Temp sensor 4 - Not used.
Temp sensor 5 - Not used.
Temp sensor 6 - Intake (directly below and in front of boards).
Temp sensor 7 - Not used.
Temp sensor 8 - Not used.
Reading of 10 on temp sensor - sensor most likely disconnected.

11.* Using pwr_util

There are certain situations that warrant powering certain areas of the system up without going through the entire diaginit process. This procedure can be followed to circumvent the diaginit and induce the component to complete the power applied process.

- a) Bpc chk interlock
- b) Reset uart
- c) Bpc reset
- d) Init bpc
- e) Offline bpc
- f) F/W revision
- g) F/W Download
- h) Bay Config chk
- i) Cop Read
- j) Cop Write
- k) Bay Power chk
- l) Power On
- m) Power Down
- n) Send Status
- o) Set Voltage
- p) Set temp
- r) busses off
- t) Transparent
- q) Quit
- ?) Print Menu

To reproduce the diaginit process it is necessary to execute the following commands in order:

- 1) c (Reset the BPC) Assert the reset line on a specified BPC. This forces the set of PPCs connected to this BPC into reset also.

C3800 Troubleshooting Guide

- 2) d Reinitialize the BPC and force it to look for interlock with its PPCs. This operation is normally done when powering on the system as the first step in establishing communications between SPU and BPCs.
- 3) h Force the BPC to poll its PPCs and check that the configuration in the bay associated with this BPC is valid. A block of data, called the bay status message, is reported to the user after this operation which shows the status of the bay and each PPC connected to it. The BPC updates its internal information on the status of each PPC and the BPC environment based on this information.
- 4) k Force the BPC to poll its PPCs and check that the power supplies in the bay associated with this BPC is valid. A block of data, called the bay status message, is reported to the user after this operation to show the status of the power system. The BPC updates its internal information on the status of each PPC and the BPC environment based on this information.
- 5) l Apply power to a board's busses. This will not bring up a board that has lost its 300 volt supply.

The above procedure will permit a total power configuration of the selected system components.

A status of the powered up components can then be obtained with a "n". This will yield the following display:

```

PPC Status
bay pp fw prt bkpln bkpln plt bd bd bd bd bd bd bd bay bps bus bus
cnf cnf rev id slot type typ 0 1 2 3 4 5 6 7 pwr num OK on
0|ff ff ffff ff ff ff ff ff ff ff ff ff ff ff ff f3 ff ff ff
1|00 00 0110 00 06 02 01 01 00 00 00 00 00 00 00 00 00 00 00 00
2|00 00 0110 00 07 02 02 02 00 00 00 00 00 00 00 00 01 00 00
3|00 00 010d 00 08 02 06 06 00 00 00 00 00 00 00 00 01 00 00
4|00 00 0110 01 09 03 06 06 00 00 00 00 00 00 00 00 04 00 00
5|00 00 0110 01 0a 03 02 02 00 00 00 00 00 00 00 00 04 00 00
6|00 00 0110 01 0b 03 01 01 00 00 00 00 00 00 00 00 05 00 00
7|ff ff ffff ff ff ff ff ff ff ff ff ff ff ff ff f3 ff ff ff
Temperature Status
f/w rev temp stat temp_0 temp_1 temp_2 temp_3 temp_4 temp_5 temp_6 temp_7
0209 00 ff ff c1 ff ff ae ff ff
in_wrm_set in_hot_set out_wrm_set out_hot_set BPC err fan0 fan1 fan2
9c 82 47 3a 00 00 00 00

```

C3800 Troubleshooting Guide

12.* Isolating Crossbar Power Problems

It should be understood that for power configuration purposes, it is possible to remove individual crossbar boards and still apply power to the crossbar.

This procedure can be used when attempting to isolate failures in the power configuration portion of diaginit, that are suspected of being caused by a single board.

This can also be done in isolation of even and odd xbar power pallet failures. Or, to copmod a xbar board which can be accomplished by first entering a "d" under pwr_util and then a "h". This is sufficient to get power to the cop chips, so that copmod can be executed.

13.* CCU Temperature Sensors

It has been found that the CCU Temperature Sensors do not use the "front facing" concept when reporting temperature warnings. The following is an example of a Hot BPC Temperature message. As can be seen, the below message is reporting the problem for the RightCCU Outlet. The RightCCU Outlet is the one on your right as you are facing AWAY from the cabinet. It was verified that messages reported for the LeftCCU Outlet will be on your left facing AWAY from the cabinet.

```

+++
<Wed Nov 11 15:14:18 1992> /diag/bin/bpcwatchd(245):../bpcwatchd.c:164
Env Error (DiagER350): message received by SPU

```

```

BPC: 4 BAY: UNKNOWN
PPC: UNKNOWN SLOT: UNKNOWN TARGET: UNKNOWN
BPC MSG: REPORTED HOT BPC TEMPERATURE
RightCCU Outlet

```

```

msgid:82 length:08 source:08 nwi_uart:04 misc0:00 misc1:00 group:72 error:40
@BPC= 4 @BAY= UNKNOWN @MSGID= 82 @GROUP= 72 @ERROR= 40
@PPC= UNKNOWN @SLOT= UNKNOWN @TARGET= UNKNOWN
****

```

```

+++
<Wed Nov 11 15:14:18 1992> /diag/bin/bpcwatchd(245):../bpcwatchd.c:164
Env Error (DiagER350): message received by SPU

```

```

BPC: 4 BAY: UNKNOWN
PPC: UNKNOWN SLOT: UNKNOWN TARGET: UNKNOWN
BPC MSG: REPORTED WARM BPC TEMPERATURE ON SENSORS:
RightCCU Outlet

```

```

msgid:82 length:08 source:08 nwi_uart:04 misc0:00 misc1:00 group:81 error:40
@BPC= 4 @BAY= UNKNOWN @MSGID= 82 @GROUP= 81 @ERROR= 40
@PPC= UNKNOWN @SLOT= UNKNOWN @TARGET= UNKNOWN

```

93/02/22
07:51:00

(trngps)(mikey:throg Job: vol.8 _ate: Mon Feb 22 07:50:58 1993)

psfilter.in.23447

63

C3800 Troubleshooting Guide

CHAPTER VII (I/O)

1.* CCU Revision Levels

The VIOP must be at rev K to function on a C3800 system. This rev is downward compatible and can be used on all systems. In the future only this rev and higher will be issued through logistics.

The TLI should be at rev G, or above and the IDC must be a rev T in order to function in a C3800.

Changes have been made to the firmware that support the extended addressing use by the NIA in the C3800.

2.* C3800 IA Soft Errors

It should be understood that IA soft errors, on a C3800, are the equivalent of PIA soft errors on the C2. In essence it means that the NIA has detected a hard error form one, or more CCU's. These errors can indicate anything from a failed CCU to a dropped data bit in a Pbus transfer.

Although it is possible for the NIA to cause this failure, it is not first choice. The CCU's should always be the primary suspect, followed by the devices on the CCU. In the cases of controllers causing the problem, the procedure would be the same as for the C2.

In order to isolate the failure there are two diagnostic tests that can be useful. These tests are pb_walk and idc_con. As idc_con will only run against an IDC, it may be necessary to move an IDC into the suspected slot.

The utility pb_walk can be executed against a specific Pbus by entering:

```
pb_walk x (where x is the Pbus to be tested)
```

In addition, it may be helpful to execute the test with "ignore errors" set so that all patterns can be tested. This will help in identifying any bit patterns that are unsuccessful. If this is not done then pb_walk will always stop execution after the first error. To accomplish this, execute the following:

```
pb_walk -e 0 1 ia8 ( In this example, the test will be executed against  
Pbus's 0 and 1 on IA8 and will ignore the errors and  
run all patterns to completion)
```

Using the "-e" switch enables you to search for dropped, or shorted bits as the test runs through the sliding pattern. Individual net problems will become obvious very quickly.

It may also be helpful to loop on the test and for this a script will have to be written. An example csh script is as follows:

C3800 Troubleshooting Guide

```
#  
while (1)  
pb_walk -e 0 1  
idc_con 32 34  
end
```

The utility idc_con is included in this script as a further example. Keep in mind that this is a csh script and so, to execute will require popping a csh window on the OPUS, or enter a /bin/csh at the spu prompt.

As mentioned above idc_con can also be helpful for isolating these failures. The same "-e" switch can be used with idc_con. The CCU slot must be entered when executing idc_con, or it will not execute.

For the best chance at locating the source of the failure, it is best to execute these tests directly after a crash. It may be necessary to issue a cleanup if "clock generator busy" messages are encountered.

SST will be of no assistance in this area as there are no modules loaded for the TTL nets.

As mentioned in a previous Tech Bulletin, rslog can be very helpful in isolating the failure to a specific Pbus and , or CCU. Please see that info for more details.

3.* I/O Bulkhead Connectors

In order to save money, we will no longer ship an IDC bulkhead in each unused I/O slot, on C3800's.

From this point forward only one unused IDC bulkhead will be shipped with new systems. This extra one can be used for troubleshooting purposes and make upgrades easier. For maximum benefit, the spare will always be installed on the unused Pbus, if one exists. If not in use, this will always be on Pbus 2, or 3. This will make it far easier to move a CCU and power to this side and verify functionality.

In addition, the unused augat, XIOP slot, in bay 4 will be removed as a cost savings measure. The combined savings for these two measures will be \$1480 per system. For systems already installed, this augat can be used as a spare if deemed necessary.

These cost savings, when applied to only the existing backlog, amount to nearly \$50,000.

These measures should have little impact on the field and has significant benefits for CONVEX.

93/02/22
07:51:00

(trngps)(mikey:throg Job: vol.8 Date: Mon Feb 22 07:50:58 1993)

psfilter.in.23447

64

C3800 Troubleshooting Guide

4.* Loading tli4480

It is recommended that the diagnostic /diag/test/tli4480.t be saved, prior to loading tli software on the JP. The TLI software installation will copy a new version of tli4480.t to the SPU and this version will not execute.

After loading this software on a C3800, it is then possible to copy your saved version over that downloaded during the tli software installation.

5.* HiPPI Installation Prereq

It should be understood that to install HiPPI in a C3800 an assembly rev B CCU Power Pallet is required.

To verify that the correct power pallet is installed in the system, it is only necessary to remove the cover and examine the part number tag affixed to the front of the power pallet. If it is HiPPI compatible the label will indicate (HiPPI). If the label does not indicate this then the incorrect CCU Power Pallet is installed for HiPPI support.

6.* C3800 Pbus Architecture

It should be understood that the Pbus on the C3800 is, architecturally, identical to the Pbus on a C2, so that all characteristics apply to both.

This includes the fact that the x and y ports between vbcu and viop are reversed, as referenced to each other. This means that when moving the x cables to the y side for testing, or vice-versa, it is necessary to rotate the cables. This means that y3 will go to x1 and so-forth.

7.* io5000 Failure on C3800

It is not possible to execute diagnostic io5000.t on multiple VIOP's when installed in a C3800. This does not apply to any other CONVEX product.

When executing io5000 on more than a single VIOP, the second module will fail when it is booted during substest 200.

The temporary solution is to execute the io5000 diagnostic on a single VIOP at a time.

C3800 Troubleshooting Guide

8.* Failures related to FDDI

When running FDDI on a C3800 system, it is necessary to add the sticky bit to the file /mnt/os/drvfsd. This can be accomplished as follows:

```
chmod 4755 /mnt/os/drvfsd
```

This should be accomplished to prevent failures with vme based diagnostics and failures during boot of the type:

```
mmunit: memory initialization complete
chguid: must be run as root or convexas.
drvfsd: warning: may need to be manually killed if kernel panics
[SPU @11:22:55] Errlog started: -l /mnt/errlog
[SPU @11:22:55] <Mon Sep 21 1992>
Loading vmunix
errintd(R1.0) started, options: -h -s
mm_sniff: sniff rate: 32.14 MB/day (15.86 days/pass)
vmunix:text:1445888 data: 155648 tdata: 4096 tbss: 20480 bss: 1110016
```

After this the boot will abort. When this occurs, it is necessary to kill all drvfsd processes currently running on the SPU, except the most recent.

9.* hpi4000 Diagnostic Failure

A problem has been identified in the "hpi4000" diagnostic when running on a C38 series system.

All diagnostic substests will pass with the exception of substest 2700. The following is an example of the error that will be seen:

```
Substest 2700 0:00:00 Microsecond Counter and Line Clock
0:00:01 failed
```

```
***** Mon Aug 17 14:06:34 1992 *****
Test: hpi4000.t 1.4 Class: 2 Subtest: 2700 1.5 Count: 1 Error: 0
Failed: Microsecond Counter and Line Clock interrupt
```

```
----- Trace point: 2700.1 -----
```

Error 0x12: Line clock interrupt test

60 line clock interrupts occurred in one second

Minimum time between interrupts was 16667 uS

Maximum time between interrupts was 16667 uS

Splits array:

```
0x9ffd97 0xa03eb2 0xa07fcd 0xa0c0e8 0xa10202 0xa1431d 0xa18438 0xa1c549
0xa20663 0xa2477e 0xa28899 0xa2c9b3 0xa30ace 0xa34be9 0xa38d04 0xa3ce1e
0xa40f39 0xa4504a 0xa49165 0xa4d27f 0xa5139a 0xa554b5 0xa595cf 0xa5d6ea
0xa61805 0xa65920 0xa69a31 0xa6db4b 0xa71c66 0xa75d81 0xa79e9b 0xa7dfb6
0xa820d1 0xa861eb 0xa8a306 0xa8e421 0xa92532 0xa9664d 0xa9a767 0xa9e882
0xaa299d 0xaa6ab7 0xaaabd2 0xaaeced 0xab2e07 0xab6f22 0xabb033 0xabf14e
0xac3269 0xac7383 0xacb49e 0xacf5b9 0xad36d3 0xad77ee 0xad909 0xadfala
0xae3b34 0xae7c4f 0xaebd6a 0xae7e85
```

Test 'hpi4000.t' failed

Elapsed time: 0:03:08

***** Last command returned status 1 *****

93/02/22
07:51:00

(trngps)(mikey:throg Job: vol.8 Date: Mon Feb 22 07:50:58 1993)

psfilter.in.23447

65

C3800 Troubleshooting Guide

The reason, is the line clock is always read as 60hz in a c38 system. However, the diag. does not know this and calls a function that returns 50 in a 50hz country. Based on this wrong input, the diag will fail.

Please note that this diag. will pass on a C2.

10.* IDC related ECC errors

It has been found that it is possible for the IDC to report an unrecoverable ECC error as a recoverable error. The problem is caused by the driver and can occur at any time.

What happens is that the IDC detects an ECC error on a sector and on the reread the failure persists. This is, so through the maximum number of retries which is 10. But even though the data was not recovered in the maximum number of retries, the error message "FSC 0x102c" is still used to represent the failure. As the data was not recovered, the recoverable ECC error is the improper designation.

To determine the recoverable errors from the unrecoverable errors, it is only necessary to examine the retry count. If the retry count is 10, or less, then the data was recoverable. If the retry count is equal to 11, then it is an unrecoverable ECC error and should be handled accordingly.

11.* VIOP Memory Pages With Ethernet

The VIOP memory page allocation list is incorrect for ethernet when loading the ethernet related patches 10.0.146, 10.1.138 and 10.2.137. These patches include a new ethernet driver which uses 17 pages instead of the 11 pages that are consumed with the old driver.

In addition, this new ethernet driver is not tunable. As this driver only executes a single process instead of the four used by the previous driver, any effort to detune will result in a major loss of performance.

The past procedures for detuning the ethernet is now invalid.

This change in memory usage is the reason that VME configurations are running out of VIOP memory.

12.* Installing an ITC

There has been some confusion regarding the installation of the ITC on the C2 and C3 series systems.

The ITC is actually a respun IDC, part number 410-002228-200. In fact the data base will even consider this an IDC, even if used as an ITC.

C3800 Troubleshooting Guide

The only support for ITC, on a C2, or C3400, is contained in the Diagnostic I/O release 1.1. It will not help to add the part number to the DB_cop file, as it will still fail the scn_link.

For a C3800 with 3.0.0.6 diagnostics, it is possible to add this to the data base. It is included in the database in future releases of diagnostics.

13. VIOP Pages

In a previous tech bulletin (v3w46, article 2) a mistake was made in the text size requirements for 10.x. The proper values are given below. Sorry for the inconvenience this may have caused.

In addition, it has come to light that the SCSI related patches, 10.1.128 and 10.2.128 use one extra page. This means that the MTC-201 and MTC-202 controllers will use 15 and 16 pages, respectively, when this patch is installed. This patch is included in 10.1.2 and production 10.2.

EGOS	9	
text data	55	
text data w/Ultra	60	
FDDI	7	
ACM-201	14	
DKC-203	4	Add one page for each disk over one
DKC-204	4	Add one page for each disk over one
LAN-007	11	
tuned=3	9	
tuned=2	7	
tuned=1	5	

*NOTE: With the ethernet patch installed, the memory usage becomes 17 pages and is no longer tunable, as only a single process is executed in this new driver.

LAN-202	5	
LAN-204	1	
MTC-201	14	15 pages with SCSI patch
MTC-202	15	16 pages with SCSI patch
VTEC	4	

14.* Idcfmt failure on SPU

It should be understood that it is, currently, not possible to successfully execute idcfmt on the SPU of a C3800 system. It will be necessary to format all IDC drives at the ConvexOS level.

This problem has been resolved in the 3.3 Diagnostic release, which has been released and can be expected to begin shipping within 2 weeks.

C3800 Troubleshooting Guide

CHAPTER VIII
(Utilities and Registers)

1.* C3800 Utilities List

This includes a list of 48 basic utilities to be used and understood with the C3800. Nearly, every one of these utilities are discussed in the C3800 Troubleshooting Hints Guide. It is recommended that all Fe's with 3800 responsibilities, review this list and insure familiarization with all on the list, before they are needed.

If there are any questions concerning any of these utilities, please contact the TAC.

	System Init Utilities	

diaginit	sys_shutdown	cleanup
errintd	mminit	sysreset
initall	abd	
	Error Analysis Tools	

rslog	hard_err1	hard_err2
xbar_err	nmb_errs	dump_swip
dump_swis	display_log	dump_soft_log
hard_logger	nsp_haz	hang
icach	dcach	ptecach
rcque	mam	
	Database Utilities	

copmod	cop	cop contents
cdb_browser	fw_rev_update	ucode_rev_update
xsys_config	osc_update	diag_rev_update
	Power Tools	

pwr_util	powermon	powerdown
cpualloc		
	Diagnostics	

pb_walk	idc_con	ncutestall
sys_con	xbinteg	xc_con
sst	cpucti	xdiag
mtst		

2.* C3800 Registers

This is a list of all C3800 registers that are accessible through the SPU interface. These registers can be accessed from a dsh window with a "get".

Most of these registers are not writable, but for those that are, this is accomplished from dsh with a "put".

C3800 Troubleshooting Guide

swis_bsrc_msb	swip_bsrc_lsb	swip_bsrc_msb
swis_force_par_err	swip_force_par_err	swis_misc
swip_misc	swis_int_level_ctl	swip_int_level_ctl
swis_int_stat	swip_int_stat	swis_int_ena
swip_int_ena	swis_force_int	swip_force_int
key_switch	nwi_data_loop_back	ncu_addr_loop_back
swis_master_ena	swip_master_ena	swis_fpga_reset
swip_fpga_reset	swis_prog0	swis_prog1
swip_prog0	swip_prog1	swip_prog2
swip_prog3	uart_model	uart_mode2
uart_stat		
uart_clk_sel	uart_cmd	uart_rx_hld
uart_tx_hld	uart_input_chng	uart_aux_cnt1
uart_int_stat	uart_int_mask	init_uart_cnt_msw
uart_cnt_msw		
init_uart_cnt_lsw	uart_cnt_lsw	uart_input_port
uart_output_cfg	uart_start_cnt	uart_stop_cnt
ncu_misc_test_cnt1	ncu_xfer_cnt	main_mem_addr
mem_test_even	mem_test_odd	ncu_data_loop_back
mach_serial_number	sys_int_vec	ncu_int_stat
ncu_int_ena	ncu_err_log	clk_cmd_stat
cmd_ena1	cmd_ena2	clk_freq_cnt11
clk_freq_cnt12	clk_freq_cnt13	dis_clk1
dis_clk2	init_burst_cnt_msw	init_burst_cnt_lsw
osc_freq	phase_mon	burst_cnt_msw
burst_cnt_lsw	scn_cmd_stat	err_loc
scn_cnt	io_addr	scn_out_buf
scn_in_buf	scn_mask_buf	scn_comp_buf
soft_log_ccu_cnt1	scalar_halt	scn_cnt1_ena
scn_cnt1_ena2	hard_err1	hard_err2
hard_err_mask1	hard_err_mask2	run_ena1
run_ena2	nmb_cnt1_stat	read_rbe_count
clk_cmd_shdw	scan_mem_err_log	hw_hung

3.* C3800 APR Utility

The new utility, APR (Automatic Processor Recovery), will soon go into field beta test at selected sites. With this in mind, the following explanation of operation and functionality is offered to prepare the field for its use:

APR prerequisites are as follows:

- 1) 10.2 OS
- 2) FMI 93 (backplane upgrade and copmod)
- 3) Rev H, or above NSP
- 4) rev 2.0 SPU Unix
- 5) rev 3.0.0.6 Diagnostics
- 6) rev 3.0 SST

The intent of APR is to allow a single C3800 processor to be automatically disabled on a hard error. Other heads in the complex will continue to function uninterrupted.

93/02/22
07:51:00

(trngps)(mikey:throg Job: vol.8 Date: Mon Feb 22 07:50:58 1993)

psfilter.in.23447

67

C3800 Troubleshooting Guide

Because of the data base modifications, it will be possible to run APR on systems where all heads are not at sufficient rev. In this case only heads at rev and with backplanes properly upgraded will be re-enabled on failure.

The user process running on that head will be terminated and a core file created. All other processes will be unaffected. If the user process is executing a system call on the failed head, then the process will be signaled and continue to execute normally. Again, only process executing in ring 4 are recoverable.

If the kernel is executing on the failed head then the system will panic. The panic will be:

```
ConvexOS:FATAL ERROR:(Sched 8825)irrecoverable hard error: context jump not set
```

The object of APR will be to deallocate the head, execute SST on the failed head and if it passes, return the head to operation. All options as to number of failures allowed and whether tests are run are fully selectable by the Convex Field personnel.

The heart of APR will be the `cpu_monitor` daemon, which will monitor the state of the processors on a regular basis. The default will be 60 second intervals. The `cpu_monitor` daemon will be started in the `rc.local` file.

The `cpu_monitor` daemon will be programmable from the file `"/etc/cpu_monitor.config` file. This file is displayed below:

```
AUTO_REENABLE = 0; # 0 Indicates no auto re-enable
SST_ENABLE = 1; # 1 indicates run SST before re-enable. If the head
                # fails SST then the head will not be re-enabled.
FAILURE_COUNT = 0; # 0 indicates that unlimited errors will result in
                # restart. Any other value will indicate number of
                # retires in the below time period.

FAILURE_COUNT_TIME_LIMIT = 24 # Indicates time limit for failure count.
```

Additions have also been made to the CDB to support APR:

```
enable_cpu_harderrs - 0 will allow APR functionality, 1 will enable
                    # normal error processing.

cpu_hard_err_sniff_period - Used by cpu_monitor daemon to control monitor
                    # interval (In seconds).

dynamic_cpu_harderrs_n - Indicates if CPU and backplane for each head
                    # support APR.

cpu_automatic_realloc_n - Specifies whether to automatically re-enable
                    # failed head.
```

The backplane modification will consist of a single net from the NVP to the NSP which will inform the NSP in case of a NVP hard error. The backplane must be cop modified to indicate the presence of this net before APR can function.

C3800 Troubleshooting Guide

To determine if a head is APR capable it is, only, necessary to perform:

```
cdb_get dynamic_cpu_harderrs_n, where n is the CPU number
```

A value of "1" indicates that the head is hardware capable.

A new switch is available, in `xsystest.config`, to allow the enabling and disabling of APR/CPU hard errors.

The utility `cpuconf` `<-e, -d>` can be used to disable a head from the OS complex. When the head is enabled, control store load and `sysreset` will be automatically performed on the individual head.

Execution of `sst` will be controlled by the file `/diag/hw/cpualloc.test`. This file will be used to execute `sst` on disabled heads, in the form:

```
cpualloc.test n, Where n is the CPU to be tested.
```

This will force execution of SST on the effected NSP and NVP and indicate whether the test passed, or failed.

It is also possible to power the indicated head off and replace and execute `diaginit` on the replaced head while the system is in OS.

As explained in earlier in this article, in the C3800 Troubleshooting Hints, APR (Automatic Processor Recovery) has been introduced with the release of 10.2 OS and 3.0 Diags. With this release will be some new utilities, as well as changes to existing utilities, to support this capability. Please review this article along with the previous document for the most thorough understanding of this subject.

The command `'cdb_get'` can be used to retrieve keyword info from the configuration data base. For example, a `'cdb_get dynamic_cpu_harderrs_0'` will retrieve this entry. As stated in the other document, a 1 in this register indicates that APR is enabled on that head. The command `'cdb_get'` does not accept wild cards and will only display one entry at a time. A more convenient tool would be `'cdb_browser'` and query for an entry like `'dynam*'` which would return all 8 entries.

The utility `'udiaginit'` will now check the assembly rev of the board and the backplane to insure that the head is APR capable. If the rev of the NSP is above H, or greater and the rev of the backplane is B, or greater, then `diaginit` will set the `dynamic_cpu_harderrs` entry for that head to a 1.

```
*****NOTE***** If the backplane is not copped (FMI 93) then diaginit
                    # will set the dynamic_cpu_harderrs to 0 and thus prevent
                    # APR functionality.
```

The `'cpualloc'` utility now works to enable and disable a head online. This utility has 2 switches; `-d` which will disable a head and `-e` which will enable the head. For example: `'cpualloc -e 0'` will enable and download ucode to head 0. This utility is a SPU level function.

The utility `'cpuconf'` has been around for a while and has been used to remove a head from the scheduler at the OS level. Now `'cpuconf'` calls `cpualloc` to allow physical removal and replacement of a head in OS. This utility utilizes the same switches as `cpualloc`.

C3800 Troubleshooting Guide

A new utility is available on the spu for running sst against the specific head. This utility is 'cpualloc.test' and when followed by a cpuid will execute SST against the NSP and NVP in the specified head. This will allow a shortcut method to execute SST on the head, rather than having to run through the menus and setup.

The entry 'enable_cpu_harderrs' is used to override, or allow APR to function. Although this is an entry in the cdb, a switch is provided in the xsys_config display for setting this entry. When on APR will not function, regardless what other registers indicate. For operation, this entry should be left off in xsys_config.

As a reminder APR will not function with a single head enabled.

It will be possible to execute 'cpuconf -d n' on a specific head to remove it from the complex and then execute 'spucmd /diag/hw/cpualloc.test n' to run SST on the entire head. It would then be possible to return the head to the complex, by means of the utility 'cpuconf'.

It is not recommended to execute any diagnostics on a downed head, other than SST, because the majority of diagnostics require the crossbar. If attempted, diagnostics can result in a system crash.

It is perfectly acceptable to execute diaginit while the system is up. This will only affect the head that has been removed and will have no impact on the remainder of the complex.

The only restriction is that only one process can be performed on the downed head at any one time. Otherwise it is possible to execute SST, or cpualloc for download of microcode. An inital, or mminit cannot be performed, as these utilities are not restricted to a specific head.

It should further be understood that APR and options are a boot time parameter and cannot be changed on the fly.

****WARNING****

In addition, it is possible for APR to fail in the act of sorting out the failure mode and thus be unable to clear the error. This situation can result in a crash, but the head will still be removed from the complex. This means the system will reboot short one head. To counter this possibility it is recommended that the system be checked for available heads when rebooting.

This scenario will, also, take place after a fatal CONVEX Unix Error.

It should be understood that on a C3800 running with APR enabled, it is not possible to manually remove a head from the complex and keep it removed. This is because the cpu_monitor utility will detect that the head has been removed and attempt to reenale it.

If it is desired to remove a head from the system complex for an extended period, by means of 'cpuconf', then the cpu_monitor utility must be disabled.

C3800 Troubleshooting Guide

****NOTE A test of APR functionality can be accomplished by disabling a head with cpuconf. If APR is functioning properly, then SST will be executed on the head and it will automatically be returned to the complex.

It should be further understood that the purpose of this utility is to monitor for downed heads and then return them to the complex. It is a very simple utility and is not involved with errintd, so it does not look for crashed heads, it is concerned with any downed head.

If it's desired to keep the head out of the complex then this will have to be accomplished prior to boot with a powerdown, or removal from OS by means of xsys_config.

In cases where the system crashes because the hard error cannot be cleared, this is usually due to a crash in the kernel. In some cases the "Fatal CONVEX UNIX Error" that should occur is sometimes omitted entirely, or listed in partial form. This occurs because errintd is expected to write to 2 files simultaneously and thus will generally omit one. This problem should be resolved in version 3.3 of diags.

4.* Debug Scripts

The following scripts, located in /diag/hw, will assist you when trying to isolate C3800 memory or NIA related problems.

nmb_errs - Reports nmb detected hard or soft errors.

nmc_test - Performs connectivity tests of all testable signals between the nmb and the nmcs.

pb_walk - PBUS loopback and functional test of the NIA.

Refer to the appropriate manpage for further information concerning these three scripts.

5.* C3800 Continuity Tests and Functions

The following is a list of continuity and their function for the C3800:

- 1) sys_con -- This utility is to test for stuck at faults between the crossbar, NCU and NIA. There is, also, a limited amount signal line short testing, using the patterns 0, F, A and 5.
- 2) xc_con -- This utility is used to test continuity of data lines between the crossbar and NSP's, NCU and NIA. This utility is not executable, except in its home directory. Which is /diag/hw/cutest. If run with no arguments will only execute on the NCU.

It can be run on the other devices by using the following switches: p0-p7 (for NSP head#) and i8 (for NIA).

- 3) xbinteg -- This is a more robust version of sys_con. It is implemented by entering xbinteg e, or xbinteg o. This will execute the test on either the odd side, or the even side of the crossbar.

93/02/22
07:51:00

(trngps)(mikey:throg Job: vol.8 Date: Mon Feb 22 07:50:58 1993)

psfilter.in.23447

69

C3800 Troubleshooting Guide

4) idc_con -- This verifies continuity of the Pbus between the individual IDC and the NIA. Will use bit by bit transfers of data to memory through the NIA. Executes the following subtests:

- a) idc_init
- b) setup_piga
- c) idc_to_mem_test
- d) idc_to_nia_test
- e) idc_interrupt_test

Executed with idc_con <ccu#>

5) pb_walk -- This is a Pbus loopback and functional test of the NIA. It tests continuity as idc_con, plus the Pbus state machines. In addition this utility tests all CCU's and is not restricted. The first 3 subtests are executed on all CCU's simultaneously and the 4th is executed sequentially. The -e option can be used to ignore a failure and continue. This also works for idc_con.

The subtests are as follow:

- a) loopback
- b) pbi_test_NOP
- c) pbi_test_WRITE
- d) mbp_test

These are all scan based tests and included as modules of SST.

6.* Memory Soft Error tools

There are two tools that can be used to isolate memory soft errors. These are "dump_soft_log" and "mam". The "dump_soft_log" tool will allow you to retrieve soft errors that have occurred and "mam" will allow verification odd the address, as well as, supply additional information.

Currently IA and SP soft errors are recorded only on the console screen and in the display_log. So, if you are not setting in from of the console when one of these soft errors occur, the only way you will be aware of this occurrence is by reviewing the display_log.

This problem will be corrected in the diagnostic release 3.1. At that time you can retrieve the information with a "dump_soft_log".

To complement this utility, there is a cleaning utility for removing these soft errors. This utility is /diag/hw/soft_clean. This will remove all soft memory errors and reset the pointers for dump_soft_log.

C3800 Troubleshooting Guide

spu> dump_soft_log

LOG STARTED: Jan 00 00:00 1900
TOTAL FAILURES: 21
FAILED DEVICES: 13
ERRORS NOT LOGGED: 0

M	B	S	I	T						
B	DEVICE	T	K	FIRST FAIL	LAST FAIL	ADDR	#FAILS			

LATEST FAILURE:

1e Z032A4 20 0 Feb 05 06:48 1992 Feb 13 07:44 1992 leec9ef8 000002

EARLIEST FAILURE:

0e U002B1 29 0 Jan 11 00:46 1992 Jan 20 02:56 1992 003de868 000006

LOGGED FAILURES:

0e U002B1 29 0 Jan 11 00:46 1992 Jan 20 02:56 1992 003de868 000006
1e Z032A4 20 0 Feb 05 06:48 1992 Feb 13 07:44 1992 leec9ef8 000002

spu> mam 4 leec9ef8

EVEN REQUEST

Board 16#1
Bank 16#f
Row 16#3
row_add 16#6a6
col_add 16#7d6

The failing nmc for this entry is fe on nmb1.

The following is an explanation of the above "mam 4 leec93f8" entry.

mam = script name

4 = number of memory boards in the system.

leec93f8 = the physical address taken from the ADDR column of the dump_soft_log print out.

93/02/22
07:51:00

(trngps)(mikey:throg Job: vol.8 _ ate: Mon Feb 22 07:50:58 1993)

psfilter.in.23447

71

C3800 Troubleshooting Guide

This can cause some confusion, as the the cdb is not updated, except as a result of a power up. This means that after executing copmod, the changes will not be reflected in the cdb until power cycling the system. This means that copmod changes will not, necessarily, appear in the database when first examined.

This should not be interpreted as a failure of the copmod, but as a functional issue with the database.

10.* Unsupported Utilities

The following 3800 diagnostic utilities are no longer supported:

- 1) sys_con
- 2) xbinteg
- 3) xc_con
- 4) ncutestall

As many of you have already discovered, these utilities have many bugs and get worse with each release. All of the connectivity tests have been incorporated into spu4000 and ncutestall has been replaced with the diag cu4000.

The utilities will be removed from the 3.2 diagnostic release, to avoid confusion.

11.* APR failure to clear Harderr

There have been some isolated instances of APR failing to clear a hard error on a C3800 and prevent the system from crashing. There are two variations of the problem and are detailed as follows:

- 1) The first failure is a system hang during execution of the hard logger. There is no other indication, but it cannot be cleared, except by reboot.
- 2) The second variation is that the hard error cannot be cleared and the system crashes and executes an osclean. An example is below:

```
HardLog End (DiagER483): Hard_logger completion
```

```
****  
ERRINTD: Checking if OS has cleaned up CPU  
ERRINTD: Re-enabling local enables  
ERRINTD: Awaiting the cleanup of a DEAD CPU  
Unable to clear hard error on CPU 1 - calling OSCLEAN.
```

It should be understood that the instances of these problems have been rare and is not understood. It is possible that the failures are due to the hard error, or some other hardware failure, but until more information is collected this is a mystery.

In order to perform initial analysis of the failures, it is requested that the following script be loaded on each SPU and executed directly following one of these events.

C3800 Troubleshooting Guide

```
#!/diag/bin/dsh  
#script to dump lower 4k of memory.  
#  
cleanup  
cleanup  
cleanup  
  
filext='date +%m%d%y%H%M'  
cd /mnt/os  
fprint "0,0x200/1\n\Sq\n" | jpd -n 2>&1 > /sst/apr.out.${filext}  
echo "" >> /sst/apr.out.${filext}  
  
#  
# Grab the vmunix file that is in /mnt/os and tar that together with  
# the apr file. So that we have a single file to get.  
#  
  
tar -cf /sst/apr.out.${filext}.tar vmunix -C /sst apr.out.${filext}  
cd /sst  
compress apr.out.${filext}.tar  
exit 0
```

The purpose of this script is to dump the first page of memory and save the contents of the pc. This is all appended to the vmunix, compressed and saved in a file. This file should then be forwarded to the TAC for analysis.

12.* CXTS - Convex Xpert Troubleshooting System

For a couple of years now, CONVEX has been developing an expert system based diagnostic tool, for the C3800. This product is called CXTS (CONVEX XPERT TROUBLESHOOTING SYSTEM). Previously this product was known as CATALYST, but has undergone this name change.

This product has made its way to alpha test (in-house) and will soon be going into beta test. The beta test sites have already been selected, so if you are not aware of it, then your particular site was not included.

The purpose of CXTS is to use its data base and rules to make decisions and suggestions to be followed after a system event. In many cases, its operation will be totally automatic. In previous tech bulletins and by fax, rules and flow charts have been sent to the field, detailing exact steps to take for specific extractor failures. These rules were part of CXTS and should give a good insight into the overall intent.

93/02/22
07:51:00

(trngps)(mikey:throg Job: vol.8 Date: Mon Feb 22 07:50:58 1993)

psfilter.in.23447

72

C3800 Troubleshooting Guide

The following modules are part of CXTS:

- System picture files (detailed pictures of system components)
- Online documentation (Includes removal and replacement instructions)
- Document viewer
- User Interface (cxts_ui)
- daemon (cxts_rt)
- System configuration file (SCF)
- History Database
- Journal files
- cxts_server
- interface for errlogd
- SPU daemons
- System, I/O and scan based diagnostics

It is currently anticipated that CXTS will be available as a production release during Q193. During the next few weeks more detailed information will be made available about this product. So that by the time of its release, everyone should be aware of its functionality and operation.

The current prerequisites for CXTS is 3.2 diagnostics and 3.1.1 SST.

13.* Crashdump -H problem

In order to execute a crashdump -H on a C3800, from /mnt/os, it is necessary to add /diag/hw to the PATH entry in /mnt/os/crashdump.

A standard crashdump that does not call hwdump will execute from /mnt/os, but if a hwdump is required, then it will be necessary to add this to the path, or execute crashdump from /.

An example of the current path as seen in the crashdump file:

```
#!/bin/sh  
  
PATH=../mnt/os:/bin:/mnt/bin  
export PATH
```

14.* Head offline on reboot

It should be understood that if a head has been removed from the OS configuration, by the APR function, then this head will be off-line when the system is rebooted. This is because the current process clears the cpu_os_req field, for the head, in the database.

This problem will be corrected with the next diagnostic release, which is due in early January.

In the meantime, the safest solution is to insert a cpuconf at the bottom of the .login file, so that on a reboot the available heads will be displayed, which should remind the customer to re-enable the head.

C3800 Troubleshooting Guide

15.* ARP Start Script

In order to initiate the apr process and start the cpu_monitor daemon, it is important that the following script be inserted into the bottom of the rc.local file. Any mistakes in this entry will lead to the process not starting and therefore not executing APR.

```
if [ -f /etc/cpu_monitor -a -f /etc/cpu_monitor.config ]; then  
    /etc/cpu_monitor &  
    echo -n 'cpu_monitor'  
fi
```

Again, any inaccuracy in the entry will lead to cpu_monitor not starting. There will be no error printed to indicate this. It can only be verified by monitoring the message at boot, or executing a ps.

93/02/22
07:51:00

(trngps)(mikey:throg Job: vol.8 Date: Mon Feb 22 07:50:58 1993)

psfilter.in.23447

73

C3800 Troubleshooting Guide

CHAPTER IX
(Troubleshooting)

1.* NVRF Hard Error Explanation

As many of you have already noticed the type of hard crashes being seen on C3800's has begun to change recently. This appears primarily due to the fact that the purge ram problems are being brought under control and there is opportunity for a new situation to present itself.

The new hard errors that are being seen a great deal are referred to as "NVRF1" and "NVRF2". The "NVRF1" crash is the more conventional error and presents with multiple gate arrays reporting parity errors from the Vector Unit. These reporting components will generally consist of members of the add pipe, or multiply pipe and can include all functional units. This will result in a list of from 5 to 16 hard errors on the same vector processor. It is also possible for a parity error to be indicated in an NVRF register, or not. A parity error will appear as an * by the NVRF that had the error.

An example of this type of error can be seen below:

```
+++>
<Sat Apr 25 15:37:58 1992> /diag/bin/hard_logger.exc:./hard.c:125
SW Error (DiagER368): Hard Error Register Contents
```

```
detected_harderrors[0] = 00000100 detected_harderrors[1] = 00000041
enabled_harderrors[0] = 00005fff enabled_harderrors[1] = 0000007f
```

Hard error Summary

HARDERROR #	BOARD TYPE	PORT/SIDE	EXTRACTOR
0	VP_TYPE	2	nmisc_m.ybus_par_err

```
vp2:nmisc_m.ybus_par_err = 1
Parity error script for ybus received by NMISC_M gate array.
Register name Ring value Board signal name
-----
ybus_data<63:0> 00 00 00 00 02 00 00 00 M3_OPO_DAT<63..0>
ybus_par<0:7> 1 1 1 1 1* 1 1 1 M3_OPO_PAR<0..7>
* indicates parity error
Ring Nibble Driving board signal
NVRF# value Parity M3_OPO_DAT M3_OPO_NIBBLE_PAR
-----
NVRF0 0 1 <63..60> <0>
NVRF2 0 1 <55..52> <1>
NVRF4 0 1 <47..44> <2>
NVRF6 0 1 <39..36> <3>
NVRF0 0 1 <31..28> <4>
NVRF2 0 1 <23..20> <5>
NVRF4 0 1 <15..12> <6>
NVRF6 0 1 <7..4> <7>
```

C3800 Troubleshooting Guide

HARDERROR #	BOARD TYPE	PORT/SIDE	EXTRACTOR
1	VP_TYPE	2	nfad_m.ybus_par_err

```
vp2:nfad_m.ybus_par_err = 1
Parity error script for ybus received by NFAD_M gate array.
Register name Ring value Board signal name
-----
ybus_data<63:0> 00 00 00 00 02 00 00 00 M3_OPO_DAT<63..0>
ybus_par<0:7> 1 1 1 1 1* 1 1 1 M3_OPO_PAR<0..7>
* indicates parity error
Ring Nibble Driving board signal
NVRF# value Parity M3_OPO_DAT M3_OPO_NIBBLE_PAR
-----
NVRF0 0 1 <63..60> <0>
NVRF2 0 1 <55..52> <1>
NVRF4 0 1 <47..44> <2>
NVRF6 0 1 <39..36> <3>
NVRF0 0 1 <31..28> <4>
NVRF2 0 1 <23..20> <5>
NVRF4 0 1 <15..12> <6>
NVRF6 0 1 <7..4> <7>
```

HARDERROR #	BOARD TYPE	PORT/SIDE	EXTRACTOR
2	VP_TYPE	2	nmul.ybus_par_err

```
vp2:nmul[1].ybus_par_err = 1
Parity error script for ybus received by NMUL[1] gate array.
Register name Ring value Board signal name
-----
ybus_data<63:0> 00 00 00 00 02 00 00 00 M3_OPO_DAT<63..0>
ybus_par<0:7> 1 1 1 1 1* 1 1 1 M3_OPO_PAR<0..7>
* indicates parity error
Ring Nibble Driving board signal
NVRF# value Parity M3_OPO_DAT M3_OPO_NIBBLE_PAR
-----
NVRF0 0 1 <63..60> <0>
NVRF2 0 1 <55..52> <1>
NVRF4 0 1 <47..44> <2>
NVRF6 0 1 <39..36> <3>
NVRF0 0 1 <31..28> <4>
NVRF2 0 1 <23..20> <5>
NVRF4 0 1 <15..12> <6>
NVRF6 0 1 <7..4> <7>
```

C3800 Troubleshooting Guide

HARDERROR #	BOARD TYPE	PORT/SIDE	EXTRACTOR
3	VP_TYPE	2	nmul0.ybus_par_err

```

-----
vp2:nmul[0].ybus_par_err = 1
Parity error script for ybus received by NMUL[0] gate array.
Register name          Ring value          Board signal name
-----
ybus_data<63:0> 00 00 00 00 02 00 00 00 M3_OPO_DAT<63..0>
ybus_par<0:7> 1 1 1 1 1* 1 1 1 M3_OPO_PAR<0..7>
* indicates parity error
Ring Nibble Driving board signal
NVRF# value Parity M3_OPO_DAT M3_OPO_NIBBLE_PAR
-----
NVRF0 0 1 <63..60> <0>
NVRF2 0 1 <55..52> <1>
NVRF4 0 1 <47..44> <2>
NVRF6 0 1 <39..36> <3>
NVRF0 0 1 <31..28> <4>
NVRF2 0 1 <23..20> <5>
NVRF4 0 1 <15..12> <6>
NVRF6 0 1 <7..4> <7>

```

The second error, referred to as NVRF2 appears as a slightly different fault. The reason it is referred to as NVRF2 is because it is currently believed that this error is actually caused by the same event that generates the NVRF1 failure. The difference being that the data has moved down the line some and is actually picked up on the NSP and is displayed as a VX_DATA parity error.

Little is currently understood with this failure, but with the release of v1.1.2 of the diagnostics more should become quickly known.

An output display of this error follows:

[SPU @09:39:12] <Mon Apr 27 1992>

<Mon Apr 27 12:38:13 1992> /diag/bin/hard_logger.exc:./hard.c:125
SW Error (DiagER368): Hard Error Register Contents

```

detected_harderrors[0] = 02000000 detected_harderrors[1] = 00000041
enabled_harderrors[0] = 3f005000 enabled_harderrors[1] = 0000007f
****

```

C3800 Troubleshooting Guide

Extractor Name: xrt.rtn_par_err_sp6

Port: 0

HARDERROR #	BOARD TYPE	PORT/SIDE	EXTRACTOR
0	SP_TYPE	3	ndp.yuvc_vx_err

```

-----
sp6:ndp0.yuvc_par_err = 2 \
sp6:ndp1.yuvc_par_err = 2 | These should all have the same value
sp6:ndp2.yuvc_par_err = 2 /
Parity error script for Vx bus received by NDP0, NDP1, and NDP2 gate
arrays.
Register name          Ring value          Board signal name
-----
vx<> 00 00 00 00 00 04 00 00 VX_DATA<63..0>
vx<> 1 1 1 1 1 1* 1 1 VX_PAR<0..7>
* indicates parity error

```

HARDERROR #	BOARD TYPE	PORT/SIDE	EXTRACTOR
1	SP_TYPE	3	nrc.sal_miss3_stop_in

```

-----
sp6:nrc0.par_err_stop_in = 1 \
sp6:nrc1.par_err_stop_in = 1 | These should all have the same value
sp6:nrc2.par_err_stop_in = 1 /

```

Parity error script for sal_miss1 bus received by NRC0, NRC1, and NRC2 gate arrays.

Register name	Ring value	Board signal name
sal_miss3<>	0f 2d 33 48	SA1_MISS1<31..3>:SA1_ADDR2<2..0>
sal_miss3<>	1 1 1 0*	SA1_ADDR2_PAR<0..3>

2.* NDAT Error Hints

Hard Errors of the type "cu_ndat0_harderror", or "cu_ndat1_harderr" are, quite often, an indication of a failure in some other area of the system and not necessarily with the NCU. These errors can be very misleading and not easily recognized.

Typically these failures do not occur as HARDERROR # 0 and are usually accompanied by memory input staging errors, or memory control errors. It is also likely that the memory failures can complain of data from a specific head. When this occurs, if the head is other than 0 than the failure will quite likely be a connectivity failure associated with that SP. If, on the other hand, the data originated from SP0 then it will not be as clear. This is because SP0 is designated in certain undefined conditions.

C3800 Troubleshooting Guide

The utility 'xbar_err' can be very helpful in isolation of these failures. This utility will display all receiving boards that have encountered the data and where it's believed that the data originated. If memory input staging data errors are indicated in the display then the confidence will be quite high that the source is a connectivity failure with that SP, or a XS1 failure.

If the problem is XS1 related then the problem will be movable by rotating the XS1 from one side (even, odd) to the other.

Only after XBar and connectivity problems have been ruled out, should the NCU be suspected.

The reason that these failures can be so difficult is because the NDAT gate arrays donot hold the correct error state and therefore will list multiple possibilities for failure location. A sample display appears below:

```

-----
HARDERROR #   BOARD TYPE   PORT/SIDE   EXTRACTOR
-----
2             CU_TYPE       N/A         cu_ndat1_harderr
-----
NCU harderror detected
ndat[1].ndat_harderr = 1
-----
Due to an NDAT design error, the NDAT does not hold it's error state
correctly. The hard error has been detected and can be one of
the following conditions:
1) Parity error from the XBAR on the signals
   XSE_CU_WR_DAT<31..0>/XSE_CU_WR_PAR<3..0>
2) Parity error from the NADR on the signals
   L1_WR_ODAT<7..0>/L1_WR_OPAR<3>
3) Parity error from even comm register rams on signals
   CMR_RD_EDAT<31..0>/CMR_RD_EPAR<3..0>

```

In the case, above, only the XSE_CU_WR_DAT occurs external to the board. So, it should only be necessary to check these nets between the source and XBAR and the XBAR and NCU. But generally, if the connectivity problem rests on the NCU, or slot, then memory boards will not be involved.

The error analysis rules, sent in previous tech bulletins, can be very helpful in isolating this failure. As indicated, cpu test 4332 can help in further isolation of the failure.

If everything attempted appear ambiguous, or no change is seen then the failure, most likely, rests with the NCU.

3.* Diagnosing NMB Failures

Memory related input staging register failures are reasonably easy to fix if it's first understood and the proper approach is utilized.

C3800 Troubleshooting Guide

First it should be understood that the Input staging register is the first place in the data path of the NMB that the parity is checked. With this in mind, it is easy to understand that a parity error at this point generally indicates a failure external to the memory board itself.

An example of this type of error can be seen below:

```

-----
HARDERROR #   BOARD TYPE   PORT/SIDE   EXTRACTOR
-----
1             MB_TYPE       0           mb_ise_data_perr
-----
EVEN side 'ise_data' ctl_par_err detected in discrete input staging
registers by MBO. Request came from SP0 through the XS<even>.
-----
PARITY CHECK - The XBAR data is saved in lookaside registers in
the XS1 and XS0. This is the data that the XBAR
sent to the NMB for this request. Any difference
between the XBAR data and the NMB data may indicate
a connectivity problem between the XBAR and the NMB.
Otherwise the XBAR or the SP-XBAR connection is bad.
-----
Data          Scan Field
-----
***Data was not preserved in SP0***
-----
ac 00 00 3f   xs1e:wr_data_1sd2[0]<31..0>
1 1 1 1      xs1e:wr_par_1sd2[0]<0..3>
-----
ac 00 01 3f   mbs0:ise_sys.rwre_data<31..0>
1 1 1* 1     mbs0:ise_sys.rwre_par<0..3>
-----
* indicates parity error.
Note that quantities in "<>" delimiters are bit descriptors.
They are not part of the scan field name.
-----

```

This error could have just as easily occur on the odd side with mb_iso_data_perr.

As can be seen from the example, there is a lot of information displayed. The display indicates that the source of the failure is SP0. This is very valid information, except when SP0 is indicated. This is because SP0 is the default and as such will be called out when no other source is available. If the source had been SP1 then this would have been a different situation.

The next, valuable, source of information is the data display for xs1 and mbs0. This data should be evaluated for any differences. As can be seen above, there is a difference between the data exiting the crossbar <xs1e:wr_data_1sd2> and the input to the NMB <mbs0:ise_sys.rwre_data>. The difference being bit 8. This is a definite indication of a connectivity failure between the Xbar and that memory board.

If no defence exists between the Xbar and memory data, then the utility xbar_err should be used to determine the likely source of the error. In this case if the data had not changed and with SP0 called as the source a likely candidate would be XS1e, or SP0.

93/02/22
07:51:00

(trngps)(mikey:throg Job: vol.8 _ate: Mon Feb 22 07:50:58 1993)

psfilter.in.23447

76

C3800 Troubleshooting Guide

In many cases the above hard error will be accompanied by `ctl_par_err`'s, which can generally be ignored and `send_par_err`'s which is displayed below:

```

-----
HARDERROR #   BOARD TYPE   PORT/SIDE   EXTRACTOR
-----
      2       XSl_TYPE       0       xsl.send_par_err_mb0
-----
Send parity error detected by MBO from SPO in XS0/1E
  Data                Scan Field
-----
***Data was not preserved in SPO***
-----
ac 00 00 3f  xsl:wr_data_1sd2[0]<31..0>
  1  1  1  1  xsl:wr_par_1sd2[0]<0..3>
-----
ac 00 01 3f  mbs0:ise_sys.rwre_data<31..0>
  1  1  1* 1  mbs0:ise_sys.rwre_par<0..3>
-----
* indicates parity error
-----
xsl:m_err_en<0> = 1
xsl:send_par_err<0> = 1
+++>
<Wed Apr 29 18:33:56 1992>  logmsg:../logmsg.c:56
Hard Error Message from Extractor (DiagER349): logmsg: General purpose event
Source: xsl_send_par_err_mb0
@extractor=xsl_send_par_err_mb0 @board_type=xsl @port=0
****

```

This error will normally provide supporting evidence of the Input staging register error. The Xbar and memory data registers should indicate the same data as the Input staging error. Additional information will be the `xsl:m_err_en<0> = 1` and the `xsl:send_par_err<0> = 1`. These registers should support the hard error indication.

One thing to remember is that when the `m_err_en` register is set to 1, then the clocks are stopped to XS0 and XSl. This will be confirmed in the `xbar_err`. The supporting `xbar` output is shown below:

XBAR Error Logger/Identifier Version 0.7 (some testing done)

```

Halting System
System clocks halted
Checking xrte for errors
Error detected by xrte -- board has halted
Checking xrto for errors
Checking xs0e and xsle for errors
Error detected by xs0e and xsle -- boards have halted
NMB0 detected parity error on data from NSPO
Data Par in XBAR:  0x00000000 0x6      ***Parity Error***
addr= 3ffffffc cycle= 0 wr_zone= 0  ctl_par= 10 -No Parity Check-
Compare with NSP and NMB
Checking xs0o and xslo for errors

```

C3800 Troubleshooting Guide

4. Isolating System Hangs on C3800

The `scalar_halt` register can be a useful tool in establishing the hardware source of certain system hangs and halted systems, where hard information is lacking.

The `scalar_halt` register is a 32 bit register contained on the XCL and is used, exclusively to trap the first occurrence of a hard error, by monitoring the `stop_cntr` and `hw_hung` signals from each head.

The contents of this register can be examined by means of a "get `scalar_halt`" at the SPU Prompt.

An example of the returned output is shown below:

```

scalar_halt
VALUE: 16#00000000
|31 |30 |29 |28 # 27 | 26 | 25 | 24 # 23 | 22 | 21 | 20 # 19 | 18 | 17 | 16
|           Reserved           #           SCALAR HW HUNG
|
|15 |14 |13 |12 # 11 | 10 |  9 |  8 #  7 |  6 |  5 |  4 #  3 |  2 |  1 |  0
|           HALT MASK           #           HALT

```

The HW HUNG field indicates that the specified processor is in a hung state. In other words, not started execution of an instruction in a specified number of clocks. The processor is indicated from right to left, with bit 16 indicating processor 0 and bit 23 for processor 7.

The HALT field is generated from the `stop_cntr` signal and indicates the specified processor has halted and clocks have been stopped.

5.* nvrfl2 Crash Troubleshooting

It is recommended that after the occurrence of the second `nvrfl2` crash, on a C3800, that the NVP be rotated between heads. Once this error has shown itself to be repetitive on a specific head is necessary to separate the NVP and NSP for troubleshooting purposes and because some of the causes of the failure are believed timing related. It is recommend that the NVP be the board that is moved because the NVP is more robust and presents less risk in the move.

In many cases, the simple act of breaking up the NSP and NVP have resulted in a solution to the problem.

Below is an example of the failure that is being referred to:

```

detected_harderrors[0] = 02000000   detected_harderrors[1] = 00000041
enabled_harderrors[0]  = 3f005000   enabled_harderrors[1]  = 0000007f

```

93/02/22
07:51:00

(trngps)(mikey:throg Job: vol.8 Date: Mon Feb 22 07:50:58 1993)

psfilter.in.23447

77

C3800 Troubleshooting Guide

Extractor Name: xrt.rtn_par_err_sp6
Port: 0

HARDERROR #	BOARD TYPE	PORT/SIDE	EXTRACTOR
0	SP_TYPE	6	ndp.yuvc_vx_err

```

sp6:ndp0.yuvc_par_err = 2 \
sp6:ndp1.yuvc_par_err = 2 | These should all have the same value
sp6:ndp2.yuvc_par_err = 2 /

```

Parity error script for Vx bus received by

NDP0, NDP1, and NDP2 gate arrays.

Register name	Ring value	Board signal name
---------------	------------	-------------------

```

vx<> 00 00 00 00 00 04 00 00 VX_DATA<63..0>
vx<>  1  1  1  1  1  1* 1  1  VX_PAR<0..7>
* indicates parity error

```

6.* Investigating diaginit Failures

If you have a diaginit failure with entries in the display_log similar to the following, it is possible that the message queue for diaguser has filled up.

+++>

<Mon Aug 31 15:06:30 1992> /diag/bin/bpccommd:../cxmsgsnd.c:40
SW Error (DiagER128): msgsnd(2): operation failed

Line: 539 Source: ../bpccommd.c

```

Msgqid=0x00000003 Msgp=0xf7fff564 Msgsz=0x00000008 Msgflg=0x00000800
Errno=11
No more processes
****

```

To verify if this is the problem, enter "ipcs -qo" at any spu prompt. If QNUM for diaguser is near maximum (maximum is 99) then the only solution for correcting this problem, at this time, is to reboot the spu. The following is an example of a spu having an excessive number of messages active in the diaguser queue and therefore diaginit could not continue.

```

spu> ipcs -qo
IPC status from flt_7 as of Mon Aug 31 15:18:32 1992
T  ID  KEY      MODE      OWNER     GROUP  CBYTES  QNUM
Message Queues:
q   0  0x4c0509c4 -Rrw-rw-rw-  root     wheel   0        0
q   1  0x000000a5 -Rrw-rw-rw-  root     wheel   0        0
q   2  0x00000183 -Rrw-rw-rw-  root     wheel   43       1
q   3  0x420002e3 --rw-rw-rw- diaguser diaguser 792     99

```

The following is an example of a message queue taken while the system was in ConvexOS (Note: the QNUM for diaguser should normally be less than 10).

C3800 Troubleshooting Guide

spu> ipcs -qo

IPC status from flt_7 as of Tue Sep 1 13:41:35 1992

T	ID	KEY	MODE	OWNER	GROUP	CBYTES	QNUM
---	----	-----	------	-------	-------	--------	------

Message Queues:

q	0	0x4c0509c4	-Rrw-rw-rw-	root	wheel	0	0
q	1	0x000000a5	-Rrw-rw-rw-	root	wheel	0	0
q	2	0x00000183	-Rrw-rw-rw-	root	wheel	0	0
q	3	0x420002e3	-Rrw-rw-rw-	diaguser	diaguser	0	0

See the ipcs man page for further information. As usual, the diagnostics people are working on this problem.

7.* Locating Connectivity Problems

As a refresher for locating connectivity problems off the info contained in spu4000 substest 810:

The following example will illustrate the current error format of substest 810:

```

Source Signal: sp_xse.wr_data           Source Port: 3
Sink Signal:   sp_xse.wr_data           Sink Port:   3
Sink Field:    xsle:wr_data_is [3]
Failing Bit Position: 10
Expected:      00000400
Actual:        00000000

```

From the example; the source signal indicates the net in error. The sink signal is used a check on the failing net as the sink should be driven high on the next clock. The net indicates the direction of the signal. In this case the sp is the source with the xse as the termination point. The termination is always found on the second field of the net.

The source port indicates the point of origin for the failure. In the example the net would become sp3_xse.wr_data. The port value will follow physically the configuration of the system. In the case of a value "9" in this location, will mean that the xbar is the source.

The sink field is generally a register location on the board that detected the failure. In this case it is the input staging register on xsle.

The failing bit position is self explanatory. The expected value is the data pattern expected to be seen by the test and the actual indicates the results. Keep in mind that a 0 value for the failed bit will indicate a open and a 1 value indicates a short. Quite often multiple 1's will be displayed for shorted nets.

8.* ucode_pulled_hard_err analysis

The C3800 hard error "ucode_pulled_hard_err" has been showing up more often of late. This failure is not a traditional hard error and is more closely related to the "Fatal CONVEX UNIX ERROR". In fact the error is almost identical in content to the ASP104 seen on the C2 class systems. In other words, this failure can be caused by almost anything.

93/02/22
07:51:00

78

C3800 Troubleshooting Guide

When encountering this error, it's best to collect as much information as possible. This will, especially, include a dump of xbar_err. This is because the xrt's can stop during this failure and generate some unintelligible information.

One other piece of information displayed by the extractor is the uirl_upc, which is the microsequencer pc. This is equivalent to the ipc on the C2. The legitimate codes are bf4, bf6, bf8, bfa and bff, where:

- bf4 = invalid fault vector
- bf6 = rtnc frame w/FRL nonzero
- bf8 = System resource structure underflow
- bfa = pulled by diag instruction
- bff = recursive page fault

NOTE These are identical to C2 error codes.

An example is shown below:

Hard error Summary

```

-----
HARDERROR #   BOARD TYPE   PORT/SIDE   EXTRACTOR
-----
0             SP_TYPE       1
sp_npsw.ucose_pulled_hard_err
-----
spl:npsw.ucose_pulled_hard_err = 1
The microsequencer detected an unhealthy situation and asserted
a hard error to indicate this.
uirl_upc = bf8 **** System Resource Structure (SRS) Underflow
-----
+++>
<Thu Oct 8 08:27:57 1992> logmsg:../logmsg.c:56
Hard Error (DiagER349): logmsg: General purpose event
Source: sp_npsw_ucose_pulled_hard_err
@extractor=sp_npsw_ucose_pulled_hard_err @board_type=sp @port=1
****

```

9.* wredc_par_err analysis

It should not be taken for granted that "wredc_par_err" involves only the nmb. The accompanying hard errors should always be examined to determine if another source of the failure may exist. This advice should be followed for any traditional memory error. Although the error's can appear, at first glance, to be obvious, this is not necessarily true. It should be understood that extractors will execute in a predictable order and always in that order. This fact can further complicate matters.

Any hard error sequence from the hard_logger should, initially, be evaluated for the proper architectural sequence of the errors. If this is not accomplished then the wrong solution will be applied to the failure.

C3800 Troubleshooting Guide

In the harderror sequence below, the failure is actually caused by an NSP. This relationship can be seen in both harderror#1 and #2. Further evidence can usually be obtained from examining the crossbar for errors. A hint might be the existence of a iso, or ise parity error. Due to the location of the input staging register, it can suggest that the failure can be found external to the memory board.

The second hard error sequence shows a double bit rdedc error. But, because of the presence of the "iso_addr_par_err", the XS00 should be suspected.

Further evaluation should always occur when unsure of the source.

EXAMPLE 1

Hard error Summary

```

-----
HARDERROR #   BOARD TYPE   PORT/SIDE   EXTRACTOR
-----
0             MB_TYPE       0             mb_odd_bank_wredc_par_err
-----
ODD side 'bank_wredc_perr': PARITY error detected in MBO
-----
WREDC PARITY error detected by BCGA 0 in bank 0 ( => BANK_0o)
during a NOP.
-----
1             mbs0:bc[0].sys.bct10_merr_pe
1             mbs0:nmc0o_rerr
1             mbs0:nmc0o_rla_ecc_check1
43b00        mbs0:nmc0o_data
f            mbs0:nmc0o_ecc (parity)
0            mbs0:bc[0].bcga_log.bct10_cycle
600         mbs0:bc0_bank0_log_addr (phys_addr is 3000)
-----
+++>
<Thu Oct 8 04:57:46 1992> logmsg:../logmsg.c:56
Hard Error (DiagER349): logmsg: General purpose event
Source: mb_odd_bank_wredc_par_err
@extractor=mb_odd_bank_wredc_par_err @board_type=mb @port=0
****

```

93/02/22
07:51:00

79

C3800 Troubleshooting Guide

HARDERROR #	BOARD TYPE	PORT/SIDE	EXTRACTOR
1	MB_TYPE	0	mb_iso_data_perr

ODD side 'iso_data' ctl_par_err detected in discrete input staging registers by MB0. Request came from SP1 through the XS<odd>.

PARITY CHECK - The XBAR data is saved in lookaside registers in the XS1 and XS0. This is the data that the XBAR sent to the NMB for this request. Any difference between the XBAR data and the NMB data may indicate a connectivity problem between the XBAR and the NMB. Otherwise the XBAR or the SP-XBAR connection is bad.

Data	Scan Field
Data was not preserved in SP1	
00 04 3b 00	xslo:wr_data_lsd2[0]<31..0>
1 1* 1* 1	xslo:wr_par_lsd2[0]<0..3>
00 04 3b 00	mbs0:iso_sys.rwro_data<31..0>
1 1* 1* 1	mbs0:iso_sys.rwro_par<0..3>

* indicates parity error.

Note that quantities in "<>" delimiters are bit descriptors. They are not part of the scan field name.

+++>
<Thu Oct 8 04:57:47 1992> logmsg:../logmsg.c:56
Hard Error (DiagER349): logmsg: General purpose event
Source: mb_iso_data_perr
@extractor=mb_iso_data_perr @board_type=mb @port=0

HARDERROR #	BOARD TYPE	PORT/SIDE	EXTRACTOR
2	XS1_TYPE	1	xsl.send_par_err_mb0

Send parity error detected by MB0 from SP1 in XS0/10
Data Scan Field

Data	Scan Field
Data was not preserved in SP1	
00 04 3b 00	xslo:wr_data_lsd2[0]<31..0>
1 1* 1* 1	xslo:wr_par_lsd2[0]<0..3>
00 04 3b 00	mbs0:iso_sys.rwro_data<31..0>
1 1* 1* 1	mbs0:iso_sys.rwro_par<0..3>

* indicates parity error

xslo:m_err_en<0> = 1
xslo:send_par_err<0> = 1
@extractor=xsl_send_par_err_mb0 @board_type=xsl @port=1

C3800 Troubleshooting Guide

EXAMPLE 2

Hard error Summary

HARDERROR #	BOARD TYPE	PORT/SIDE	EXTRACTOR
0	MB_TYPE	4	mb_odd_bank_rdedc_multi_bit_err

ODD side 'bank_rdedc_multi_bit_err' detected in RDEDCE of MB4.

MULTI-BIT error in BCGA1 BAN1 (=> BAN_5o) detected by RDEDCE during READ operation.

ODD side BCGA error log:

1	mbs4:bc[1].sys.bctl1_rtn_merr
1	mbs4:bc[1].bcga_log.bctl1_cycle
e2d7b5	mbs4:bc1_bank1_log_addr (phys_addr is 716bda8)

ODD side RDEDCE error log:

17fefb80	mbs4:rdedc.log[1].error_data
de	mbs4:rdedc.log[1].error_ecc
ad	mbs4:rdedc.log[1].error_cmp

<Tue Oct 6 11:47:03 1992> logmsg:../logmsg.c:56
Hard Error (DiagER349): logmsg: General purpose event
Source: mb_odd_bank_rdedc_multi_bit_err
@extractor=mb_odd_bank_rdedc_multi_bit_err @board_type=mb @port=4

HARDERROR #	BOARD TYPE	PORT/SIDE	EXTRACTOR
1	MB_TYPE	4	mb_odd_bank_ctl_err

ODD side 'bank_ctl_err': Request to busy bank detected in MB4.

Request while bank busy error detected by BCGA 3 in bank 3 (=> BAN_fo). No other error state saved for this bank.

1	mbs4:bc[3].sys.bctl3_ctl_he
---	-----------------------------

+++>
<Tue Oct 6 11:47:04 1992> logmsg:../logmsg.c:56
Hard Error (DiagER349): logmsg: General purpose event
Source: mb_odd_bank_ctl_err
@extractor=mb_odd_bank_ctl_err @board_type=mb @port=4

C3800 Troubleshooting Guide

HARDERROR #	BOARD TYPE	PORT/SIDE	EXTRACTOR
2	MB_TYPE	4	mb_iso_addr_perr

ODD side 'iso_addr' ctl_par_err detected in discrete input staging registers by MB4. Request came from SP4 through the XS<odd>.

PARITY CHEC - The XBAR data is saved in lookaside registers in the XS1 and XS0. This is the data that the XBAR sent to the NMB for this request. Any difference between the XBAR data and the NMB data may indicate a connectivity problem between the XBAR and the NMB. Otherwise the XBAR or the SP-XBAR connection is bad.

Scan fields: NMB XBAR

1c mbs4:iso_sys_addr	1c xs0:addr_1sd2[4]<28..22>
0 mbs4:mbo_ctl_par<0>	0 xs1o:ctl_par_1sd2[4]
	0 xs0o:send_par_err<4>

Note that the <28..22> notation is not part of the scan field name, but represents which bits of the bus are being displayed. Bit mapping is relative to the schematic. Addr<28> from the schematic corresponds to addr<25> in the actual scan field data.

SANITY CHEC - Each BCGA in the table below should contain the same data as the discrete registers above. Data mismatch indicates a failure on the NMB. Slow (bad) timing should affect the BCGAs first.

Scan fields: NMB	Location:	BCGA
1c mbs4:bc0_sys_addr<28..22>		0
0 mbs4:bc[0].sys.ris_ctl_par_3_0<0>		
1c mbs4:bc1_sys_addr<28..22>		1
0 mbs4:bc[1].sys.ris_ctl_par_3_0<0>		
1c mbs4:bc2_sys_addr<28..22>		2
0 mbs4:bc[2].sys.ris_ctl_par_3_0<0>		
1c mbs4:bc3_sys_addr<28..22>		3
0 mbs4:bc[3].sys.ris_ctl_par_3_0<0>		

* indicates parity error.

<Tue Oct 6 11:47:06 1992> logmsg:../logmsg.c:56
Hard Error (DiagER349): logmsg: General purpose event
Source: mb_iso_addr_perr
@extractor=mb_iso_addr_perr @board_type=mb @port=4

C3800 Troubleshooting Guide

HARDERROR #	BOARD TYPE	PORT/SIDE	EXTRACTOR
3	MB_TYPE	4	mb_even_bank_ctl_err

EVEN side 'bank_ctl_err': Request to busy bank detected in MB4.

Request while bank busy error detected by BCGA 7 in bank 3 (=> BAN_fe). No other error state saved for this bank.

1 mbs4:bc[7].sys.bctl3_ctl_he

<Tue Oct 6 11:47:07 1992> logmsg:../logmsg.c:56
Hard Error (DiagER349): logmsg: General purpose event
Source: mb_even_bank_ctl_err
@extractor=mb_even_bank_ctl_err @board_type=mb @port=4
HardLog End (DiagER483): Hard_logger completion

10.* Using rcque to find data

Quite often it is necessary to pursue an intermittent system failure, on a 3800, without benefit of a supporting diagnostic failure. In these situations the only information available is that obtained from the hardlogger. This information generally consists of the extractor, port, and the data, or address involved.

After determining the source and destination of the data, or address, the solution will ultimately be based on determining the data in error. As the problem is unknown and can be anything from connectivity to a failed component, it is important that the failed data be analyzed as thoroughly as possible.

In most cases the usable information will consist of a failed byte and the parity for this byte. An example can be seen below:

HARDERROR #	BOARD TYPE	PORT/SIDE	EXTRACTOR
1	SP_TYPE	1	sp_nrc.xro_stop_in

sp3;nrc0.par_err_stop_in = 1 \
sp3;nrc1.par_err_stop_in = 1 | These should all have the same value
sp3;nrc2.par_err_stop_in = 1 /
Parity error script for xro_data
bus received by NRC0, NRC1 and NRC2 gate arrays

Register name	Ring value	Board signal name
xro_data<>	00 00 6a c1	XRO_DATA<31..0>
xro_data<>	1 1 1 1*	XRO_PAR<0..3>

*indicates parity error

C3800 Troubleshooting Guide

As can be seen, in this example, the error has occurred in byte 0 of the data, but this may not be enough information. Especially in cases where an intermittent connectivity failure is involved, this byte may involve up to 3 eponts and 4 augats, so it is important that the specific bit be determined to permit further scrutiny and monitoring. This will lead to the quickest solution.

The best tool, available, to locate the actual bit is the return control que (rcque). Often, through close examination, the failed data can be located and in these cases the lost bit can be determined.

Below is an example of the rcque output and close examination shows that the data word we are searching for is most probably located in location d of the que output. This data is 00006ac3, this would suggest that the failed bit location is at bit 1. This further isolates the failure and reduces the area to be searched. Also insure that the data matches with the extractor. IN other words, in this case, odd data is required.

[DDB]-> \$dcpu 1
[DDB]-> \$rcque

CPU1 Dump of the RETURN CONTROL QUEUE

```
----- SPL return control queue -----
ptr  dest size reg tag eo P   addr      ptr P  odd_data      P  evn_data
-----
```

00	vp-1	2	03	1e	10	8	8015ab50	00	8	2e2c3e5d	0	ae67f4d6
01	vp-1	2	03	1f	10	0	8015ab58	01	3	ae45b20f	a	2e6d28b0
02	vp-1	2	03	01	10	8	8015ab60	02	e	2d87deae	3	ae37deaa
03	vp-1	2	03	02	10	0	8015ab68	03	8	af2092ec	9	ac919d48
04	vp-1	2	03	03	10	0	8015ab70	04	3	2f4c4eb8	3	ad13e200
05	vp-1	2	03	04	10	8	8015ab78	05	c	af6c3804	5	ae489b00
06	vp-1	2	03	05	10	0	8015ab80	06	3	2f850f35	8	2dd370f8
07	vp-1	2	03	06	10	8	8015ab88	07	c	af81d54c	3	adad9ab4
08	vp-1	2	03	07	10	8	8015ab90	08	7	2f3a5372	e	2dc0cc6e
09	vp-1	2	03	08	10	0	8015ab98	09	3	aed903b4	f	2e24ac36
0a	vp-1	2	03	09	10	8	8015aba0	0a	f	00000000	7	ae442d84
0b	vp-1	2	03	0a	10	0	8015aba8	r/w> 0b	6	2f2ee84c	a	2dd6a607
0c	vp-1	2	03	0b	10	0	8015abb0	0c	8	af1592e0	a	af2af56e
0d	vp-1	2	03	0c	10	8	8015abb8	0d	f	00006ac3	5	2f42ce0a
0e	vp-1	2	03	0d	10	8	8015abc0	0e	5	ae3acd9c	c	af696b9e
0f	vp-1	2	03	0e	10	0	8015abc8	0f	0	ada80802	5	2f82f860
10	vp-1	2	03	0f	10	0	8015abd0	10	9	2d589828	8	af838532
11	vp-1	2	03	10	10	8	8015abd8	11	7	ae55d4e1	5	2f3f32aa
12	vp-1	2	03	11	10	0	8015abe0	12	e	2e849310	0	aedcdc32
13	vp-1	2	03	12	10	8	8015abe8	13	4	ae0fc8ef	f	00000000
14	vp-1	2	03	13	10	8	8015abf0	14	7	2c9c6528	r/w> 2	2f2f77bf
15	vp-1	2	03	14	10	0	8015abf8	15	0	2c1aa880	f	af1e65f5
16	vp-1	2	03	15	10	c	8015ac00	16	6	ae1bb7a8	3	2f08a066
17	vp-1	2	03	16	10	4	8015ac08	17	a	2e157d86	3	ae622bfc
r/w> 18	vp-1	2	03	1d	10	8	8015ab48	18	3	adb03f3c	2	ae045a02

C3800 Troubleshooting Guide

It should be understood that this information is only valid at the time of the event. So APR may have to be disabled in order to retrieve the correct entries.

By use of this tool, the FE is now able to concentrate, entirely, on a single bit. Without knowing the bit the search is much broader and with an intermittent may even result in failure.

93/02/22
07:51:00

(trngps)(mikey:throg Job: vol.8 Date: Mon Feb 22 07:50:58 1993)
psfilter.in.23447

82