

INGBERT GRAF
HP Munich

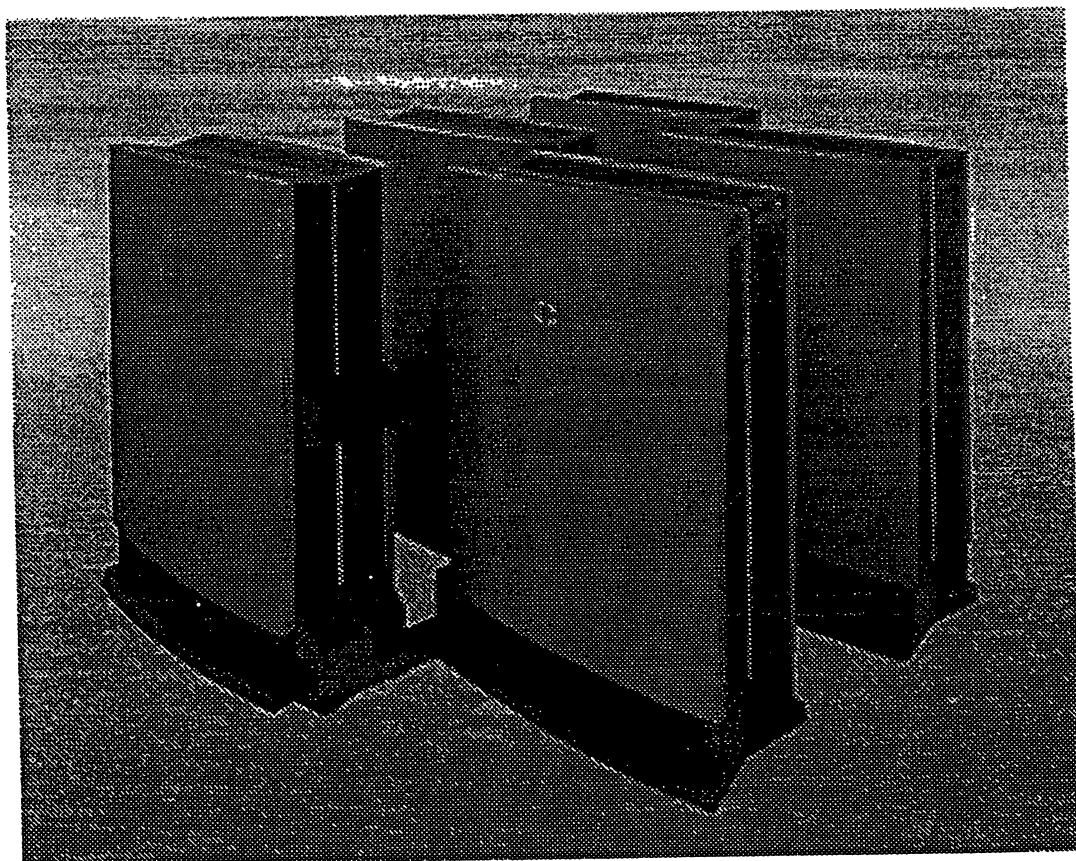
 **HEWLETT
PACKARD**
Convex Technology Center

Exemplar Series Systems Hardware Maintenance Manual

Version 8.9

Compiled and Edited
by Randy Zeid

March 14, 1996



EXEMPLAR - Hardware Maintenance Student Handbook

Copyright Material

Copyright 1995 CONVEX Computer Corporation
All rights reserved.

This document is copyrighted. This document may not, in whole or part, be copied, duplicated, reproduced, translated, electronically stored, or reduced to machine readable form without prior written consent from CONVEX Computer Corporation.

Although the material contained herein has been carefully reviewed, CONVEX Computer Corporation does not warrant it to be free of errors or omissions. CONVEX reserves the right to make corrections, updates, revisions or changes to the information contained herein. CONVEX does not warrant the material described herein to be free of patent infringement.

UNLESS PROVIDED OTHERWISE IN WRITING WITH CONVEX COMPUTER CORPORATION (CONVEX), THE PROGRAM DESCRIBED HEREIN IS PROVIDED AS IS WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. SOME STATES DO NOT ALLOW THE EXCLUSION OF IMPLIED WARRANTIES. THE ABOVE EXCLUSION MAY NOT BE APPLICABLE TO ALL PURCHASERS BECAUSE WARRANTY RIGHTS CAN VARY FROM STATE TO STATE. IN NO EVENT WILL CONVEX BE LIABLE TO ANYONE FOR SPECIAL, COLLATERAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES, INCLUDING ANY LOST PROFITS OR LOST SAVINGS, ARISING OUT OF THE USE OR INABILITY TO USE THIS PROGRAM. CONVEX WILL NOT BE LIABLE EVEN IF IT HAS BEEN NOTIFIED OF THE POSSIBILITY OF SUCH DAMAGE BY THE PURCHASER OF ANY THIRD PARTY.

CONVEX and the CONVEX logo ("C") are registered trademarks of CONVEX Computer Corporation.

ConvexOS is a trademark of CONVEX Computer Corporation

COVUE is a trademark of CONVEX Computer Corporation. COVUE products consist of COVUEbatch, COVUEbinary, COVUEedt, COVUElib, COVUenet, and COVUEshell.

UNIX is a trademark of AT&T Bell Laboratories.

X Window System is a trademark of M.I.T.

Maryland Windows is copyrighted (c) 1983 University of Maryland Computer Science Department

Printed in the United States of America

Table of Contents

Section 1 Exemplar System Overview

Basic Definitions	17
Test Station.....	17
DaRT.....	18
Node	18
Slice.....	19
CSB/ ASB	19
Node Chassis.....	19
Cabinet	20
Complex	20
Power System Descriptions	22
Power Brick.....	22
Power Brick Board.....	22
Power Concentrator Card.....	22
Power Chassis	22
Logic Systems	23
Exemplar Processors	23
MU	23
MTV.....	24
I/O Interface Cards.....	24
CPA	24
CCMC	25
XBAR.....	25
MAUI	25
SCI.....	25
Misc. Definitions	27
MOD	27

Table of Contents

Front Panel	27
OBP	27
SPP-UX	28
PVM	28
GSM	28
Section 2 Misc. Subsystems	
DaRT bus	29
DaRT and /etc/hosts	30
DaRT Cable	31
Front Panel Display	33
Typical Front Panel Messages	34
MOD Lights	36
The MOD light controller	36
MOD light cable routing	37
MOD Board Strapping	38
Part Numbers	38
Section 3 Installation	
Initial Installation	39
/etc/netlinkrc	43
Autoconfig network on boot	44
SCI RING Connection	45
SCI diagnostics	47
Multi-cabinet connection & Power sequencing	47
TOC cable Connections	48
Configurations	49
XA Single Node	50
XA 2 Node Configuration	51
CD Single Node	52
XA Single Node Multi-I/O	53
CD Multi-IO	54

CD - Max Configuration	56
XA - Max Configuration	56
Standard 6 Node Configuration	58

Section 4 The Power System

Introduction	61
Power Sequencing (ON/OFF)	62
Power Brick Descriptions	64
Power Board 1	64
Power Board 0	64
Power Board 2	64
200 Series Power bricks	65
APB Avalon Power Board	66
Power brick locations	67
Power Brick Torque	68
48V Power Supply Adjustment	68
Power Chassis Wiring	69
Power Chassis Front Panel	70
Power Chassis Concentrator Card	71

Section 5 CSB/ASB Level Functional Units

Introduction	73
CSB4 Physical Layout	73
ASB Physical Layout	77
Agent Gate Array	80
Agent Gate Array Block Diagram	81
Agent Hard Errors	82
CCMC Gate Array	83
CCMC Functional Block diagram	85
CCMC Hard Errors	86
Xbar Gate Arrays	87
Xbar Block Diagram	88
MAUI Gate Array	89

Table of Contents

MAUI Block Diagram.....	90
MAUI Hard Errors	91
SCI Gate Array	92
SCI Ring - Technical Information.....	92
SCI Block Diagram	93
T-Chip Module: SPP-1000	94
T' (T Prime) Chip: SPP-1200/1600	94
Mpp Utility board	96
MU LEDs.....	97
MU EEPROM and DRAM	97
MU Memory Allocation Map	98
Memory Tag Vortex Logic Card	99
Introduction	99
Physical memory divisions	103
Configuring Memory with CCMU.....	106
Deconfiguring Memory.....	107
Requirements.....	107
Landmarc/ Avalanche	108
 Section 6 I/O	
I/O Chassis Types	111
Boards Chassis Support Board	112
Peripheral Configurations	112
XA	113
High Performance	113
High Capacity.....	115
Power system	117
Slot Addresses.....	118
CD Configuration	119
Differential Chassis	120
Disk Drives - Strapping	121
DEC (2 GIG).....	121
Seagate (4 GIG).....	122

Dat Drive - XA/CD I/O Chassis.....	124
Dat Drive - Differential I/O Chassis	125
Ancot Card	126
IO Address Reference Sheet (nvalias format)	127
OBP Setup Parm's for IO Devices (mkmap)	129
Dat Drive Examples	129
FDDI Example	129
Section 7 Internal Communications	
The Physical data path	131
Packet Types	133
Request Packets.....	133
Response (Send) Packets.....	134
Packet Structures	136
Section 8 Diagnostics & Utilities	
Test Station Introduction	141
Software	141
The file System Tree	143
Important Test Station files	144
Backup/Restore Commands	145
SPP-UX - System Files	145
HP UX - Test Station Files.....	145
Creating a Bootable Recovery Tape - HP-UX.....	146
Bootting from the recovery tape:	146
Boot single on the Test Station	146
Software Install	147
Test Station Configuration	147
Software/ Firmware Versions.	148
MU "core"	148
Mu_firmware versions.	149
Diag_vers Command.....	150
SPP-UX Versions.....	150

Table of Contents

CCMU Camelot Configuration Management Utility	151
The CCMU Path Diagram.....	153
Important CCMU Parameters	154
Configuration Errors	155
Correct From Scratch - Using CCMU	155
Correct From File.....	156
Using Saveall and Restall.....	156
Open Boot Program	157
OBP Open Boot Prom Architecture.....	157
OBP Commands.....	158
Example of a normal OBP banner.	159
OBP standard parameters - defaults are GSM	159
OBP PVM-CTI VS GSM.....	160
Turning off OBP	161
Xconfig All	162
Cputest: T-CHIP Diagnostic	169
CST - Convex Scan Test	170
CST Connectivity Failure Message	180
CST Failures - GA Locations.....	181
do_reset	183
Sysreset Failure Message	184
Ecc_help - ECC Error Decode Utility	185
Event_logger - SPP Test Station Event Logger	187
Event_log	188
iod - I/O Diagnostic shell	189
Configuring the Disk Test.....	190
io_tc	191
load_eprom	193
Reloading the MU firmware	194
Reloading the Primary Loader	194
Reloading the Secondary Loader	195
New Versions of the load Commands	195
Saveall - Saving NVRAM Contents.	196
Restall - Restoring NVM contents.	197

mpl_map - Address Decode Tool	198
pce_util - SPP Power, Clock & Environment Utility	201
pce_util output.....	203
sn_cnsl	204
SPP Diagnostic Shell	205
Sppring CxRing and Distributed Memory Test	206
Class 1 Access Verification subtests.....	206
Class 2 Distributed Memory (Tag) Subtests	206
Class 3 Distributed Memory (Data) Subtests.....	206
Class 4 Non-Coherent Subtests	207
Class 5 Download Verification and EIR Check Subtests	207
Class 6 Intranode Coherency Subtests	207
Class 7 CxRing Internode Coherency Subtests.....	207
Class 8 Message Subtest	207
Class 9 SCI Stress Subtests	208
Class 10 Coherency Stress Subtests.....	208
Class 11 SIMM Quick Sanity Subtests	208
Class 12 Tag SIMM quick check.....	208
Class 13 Data SIMM quick check.....	208
MU power on Diagnostic	209
node_ip_set	212
verify_cables	213
verify_rings	215
Section 9 Trouble Shooting	
Gate Array Removal and Replacement	217
Removal and Replacement Parts.....	217
Cleaning	219
Maintenance	219
Orientation.....	220
OS Hangs	221
When a System Hangs:	221

Table of Contents

Identifying When the IO Board is Hung	221
Hangs - Using dump_all	223
Turning off Timouts	224
Crashdumps	224
Hard Error Debug	225
Determining the Source of a Hard Error	225
Manually Extracting Error Information	226
Agent Hard Error	228
MAUI Hard Error	231
“bio_reap_status” Message	233
Section 10 SPP-UX	
Booting SPP-UX	235
Booting from Disk	235
Boot from the Dat Drive	235
Disk Administration Under SPP-UX	236
Determining Drive Mapping	236
Mapping a Disk to a Logical Device	237
Unmapping a Disk	237
Unset Partition Flags and Descriptions	238
Set Partition Descriptions and Flags	238
Displaying Disk Partitions	239
Creating a Partition:	240
Deleting a Partition:	240
Striped Partitions	241
NEWFS	242
Mounted File Systems	242
SPP-OS FYI	243
Remote console commands	243
Passwords	243
Show Processes	243
Important Files	244

APPENDICES

<i>Appendix A</i> : Class Documentation list	245
<i>Appendix B</i> : Tools.....	246
<i>Appendix C</i> : FRU List	247
<i>Appendix D</i> : Gate Array pin-out.....	249
<i>Appendix E</i> : Quick Reference Sheet.....	254
<i>Appendix F</i> : CCMU “gets” Definitions	255
<i>Appendix G</i> : MU Status Numbers.....	266

Table of Contents

List of Figures

Section 1 Exemplar System Overview

Physical layout of the CSB	21
CSB block diagram	26

Section 2 Misc. Subsystems

Dart bus /etc/host entries	30
The DaRT bus cable	32
Front Panel Messages	34
Mod light power and data cable routing	37
MOD Board Strapping Diagram	38

Section 3 Installation

SCI XA Cabling Diagram	45
Node XA SCI Cabling Diagram	46
SCI CD Cabling Diagram	47
TOC cable connection diagram	48
XA - Single Node Configuration	50
XA - 2 Node Configuration	51
CD Single Node configuration	52
CD W/Test Station and IO chassis	52
XA Single Node Multi-I/O configuration	53
CD - Multi-IO	54
CD - I/O Chassis and cover	55
16 Node Configuration	57
XA 6 Node Configuration	58
XA Leveling Feet	59

Section 4 The Power System

Power Subsystem Diagram	63
Power Brick Locations	67
Power Supply Wiring	69

List of Figures

Power Chassis Front Panel	70
Power Concentrator Card	71
 Section 5 CSB/ASB Level Functional Units	
CSB Layout Z-Side	74
CSB U-Side Layout	75
CSB Data Paths	76
ASB Physical Layout	77
ASB U-SIDE	78
ASB Data Paths	79
Agent Gate Array Block Diagram	81
CCMC Memory Interface Block diagram	84
CCMC Functional Block diagram	85
Xbar Block Diagram	88
MAUI Block Diagram	90
SCI Block Diagram	93
T-Chip Module Block Diagram	95
MU Block Diagram	96
MU LEDs	97
MU Memory Allocation Map	98
MTV Block diagram	100
MTV Physical Layout	101
Memory configuration table	102
Node Physical Address Space Partitioning	104
Physical Memory Addressing with Ring Bits	105
SIOP Block Diagram	109
 Section 6 I/O	
The High Performance Peripheral chassis	113
The High Performance Block Diagram	114
The High Capacity Configuration	115
The High Capacity Block diagram	116
Peripheral cabinet power subsystem	117
SCSI ID's (XA Chassis)	118
CD cable diagram	119

Differential I/O Chassis	120
Disk Drive Strapping (DEC DSP3210W 2)	121
DAT Drive strapping	124
Differential to Single Ended Converter Card	126
IO Address Figure	128

Section 7 Internal Communications

Packet Word Format	137
Packet word formats (Continued)	138

Section 8 Diagnostics & Utilities

Test Station Software Overview	142
CCMU diagram	153
Xconfig Display	162
Xconfig Config Pop-up Menu	163
Xconfig Parm's Pop-up Menu	164
Xconfig Parm's/Clocks Pop-up Menu	165
Xconfig Parm's/Memory Options Pop-up Window	166
Xconfig Help Pop-up Window	167
Xconfig Help/Color Key Pop-up Window	168
CST GUI Display	171
CST Scan Ring Partitioning	172
CST Files Menu	173
CST Options Menu	174
CST Power Menu	175
CST Clocks Menu	176
CST Details Menu	177
CST Misc. Menu	178
CST Help Screen	179

Section 9 Trouble Shooting

Gate Array Removal and Replacement	218
Gate Array and Socket Orientation	220
MAUI Hard Error	231

Section 10 SPP-UX

List of Figures

List of Tables

Section 5 CSB/ASB Level Functional Units

Agent Internal Errors.....	82
Agent External Errors	82
CCMC Hard Errors	86
Maui Internal Errors.....	91
Maui External Errors.....	91
CCMU Parameter - USEMEMSIZE.....	106
I/O Boards	108

Section 7 Internal Communications

Typical packet formats (request packets).....	139
---	-----

Section 8 Diagnostics & Utilities

Gate Array Locations	181
Slice Locations	181

Section 9 Trouble Shooting

Hard Error Register - Bit Definitions.....	226
--	-----

List of Tables

Exemplar System Overview

1

1.1 Basic Definitions

In this section the “basic” building blocks of the Exemplar series systems will be covered. This includes the SPP-1000, SPP-1200 and SPP1600.

1.1.1 Test Station

- The **test station** is a dedicated work station connected to the SPP system via an ethernet connection called the DaRT bus.
 - This connection enables basic system functions to be performed:
 - ◆ loading firmware
 - ◆ running diagnostics
 - ◆ booting OS
 - ◆ and logging system level events.
 - Currently the HP-712 “Gecho” work station is being used as the SPP test station.

1.1.2 DaRT

- The **Diagnostic and Remote Testing** bus is the communication link between the SPP and the test station.
 - It is an ethernet interface that connects to all of the nodes in a complex.

1.1.3 Node

- A **node** or **hypernode**, is the basic system level unit of the SPP.
 - A fully populated node will have 8 cpus and 4 memory boards.
 - It consists of at least:
 - ◆ 2 CPUs (Tchips)
 - ◆ 2 memory boards (MTVs)
 - ◆ one system board (backplane) with the associated support gate arrays
 - ◆ 1 MU (utility board)
 - ◆ 1 SIOP (I/O board)
 - ◆ power brick boards.

Note: minimum config for a complex -2 CPUs and 2 MTVs.

1.1.4 Slice

- The node is divided up into four slices.
 - Each slice contains:
 - ◆ up to 2 cpus (PA RISC)
 - ◆ 1 cpu interface gate array (AGENT)
 - ◆ 1 memory control gate array (CCMC)
 - ◆ 1 internode communications gate array (SCI)
 - ◆ and 1 memory board. (MTV)

1.1.5 CSB/ ASB

- The **Camelot System Board** is the main board in the SPP-1000.
 - All of the system's boards plug into the **CSB**.
 - Sometimes referred to as the backplane.
 - **Figure 1-1 on page 21** is a physical layout of the CSB.
 - **Figure 1-2 on page 26** is a block diagram of the CSB.
- In the SPP-1200/1600 the system board is physically different than that of the SPP-1000. An **ASB, Avalon System Board** is used in place of the CSB.

1.1.6 Node Chassis

- The **Node Chassis** is the mechanical hardware surrounding and supporting a single Node.
 - In many contexts a Node and a Chassis are interchangeable terms.

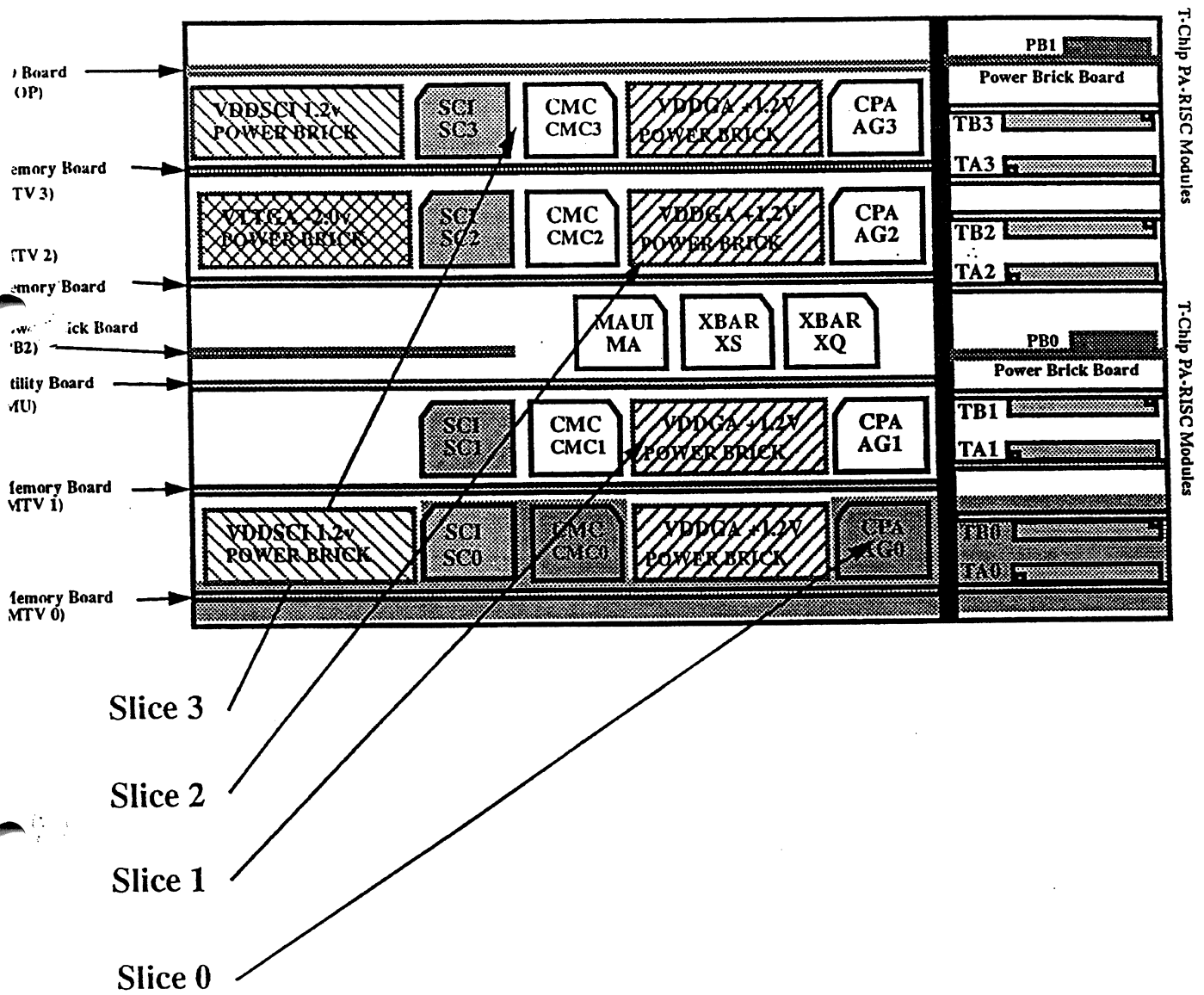
1.1.7 Cabinet

- A cabinet is the mechanical hardware surrounding and supporting one or two chassis.
 - It contains the power subsystem, indicator lighting (MODs) and side skins.
 - The XA cabinet contains 2 chassis (nodes).
 - The CD cabinet contains only one chassis.

1.1.8 Complex

- A Complex is one or more Nodes connected together as a single system.
 - Complex serial numbers:
 - ◆ XAs: 65537 thru 67599.
 - ◆ CDs: 67600 thru 69631.

Figure 1-1
Physical layout of the CSB



- The ASB differs only slightly from and CSB and will be covered in later chapters.

1.2 Power System Descriptions

1.2.1 Power Brick

- There are several power bricks used in the Exemplar.
 - The power bricks are used to convert 48VDC to the lower logic levels used by the system.
 - All of the system power is derived from a set of 48VDC power supplies, ORed together to form a single 48VDC power source.

1.2.2 Power Brick Board

- The power brick board is used to house power bricks that would not fit on the CSB, due to the lack of real-estate necessary to house them.

1.2.3 Power Concentrator Card

- A small circuit board used to monitor and control power on the SPP-XA model.
 - Attaches to the Power Chassis.
 - Not used on the CD model.

1.2.4 Power Chassis

- AC input power unit
 - AC power conditioning - line filter/ MOVs.
 - The main breaker is located on the power chassis.

1.3 Logic Systems

1.3.1 Exemplar Processors

- The **T-Chip Module** is the processor used by the SPP-1000.
 - This is an HP product, each node can have up to 8 modules - 2 per slice.
 - The **T-Chip** uses the HP PA-RISC (7100 series) 99MHZ processor with 1Mbyte of Data cache and 1Mbyte of Instruction cache.
- **T' (T-prime)** is the processor module used in the SPP-1200.
 - 120MHZ clock speed.
 - PA RISC 7200 series.
 - Cache size - 256K, both Icache and Dcache.
- SPP-1600 also uses the 7200 series processor.
 - The main difference is increased cache size.
 - The 1600 has 1Mbyte of Data cache and 1Mbyte of Instruction cache built on to the processor card.

1.3.2 MU

- The **MU (Mpp Utility)** card is the "SPU" board for the SPP.
 - It provides the interface for the test station.
 - Monitors the SPP's environmental status.
 - ◆ temps, voltage levels, etc.
 - Other run-time functions are supplied.
 - There is 1 MU in each node.

1.3.3 MTV

- The Memory Tag Vortex logic card is the memory board.
 - A node can contain up to 4 MTV's, 1 per slice.
 - Both the data and the tag rams live on the MTV.

1.3.4 I/O Interface Cards

- Landmarc - provides 4 SBUS ports and 4 Mbus ports for system I/O.
 - Only the 4 SBUS slot are usable.
 - SIOP I
- Avalanche - a newer version of the landmarc which provides for up to 8 sbus controllers.
 - SIOP II
- SIOP III - PCI support

1.3.5 CPA

- CPu Agent gate array.
 - It provides the interface between the T-Chip modules and the rest of the SPP.
 - Provides arbitration between the 2 CPUs within a slice.
 - A node can have up to 4 CPA's, 1 per slice.
 - The CPA lives directly on the CSB.
 - The agent gate array used in the SPP-1200/1600 is referred to as the APA.

1.3.6 CCMC

- **The Convex Coherent Memory Controller.**
 - Lives on the CSB and provides the MTV and the SCI (internode communication ring) with an interface to the rest of the system, via the XBAR.
 - A node can have up to 4 CCMC's, 1 per slice.

1.3.7 XBAR

- The crossbar system is made up by 2 Xbar gate arrays.
 - One request and one response gate array.
 - It provides a way for memory to be shared between slices.

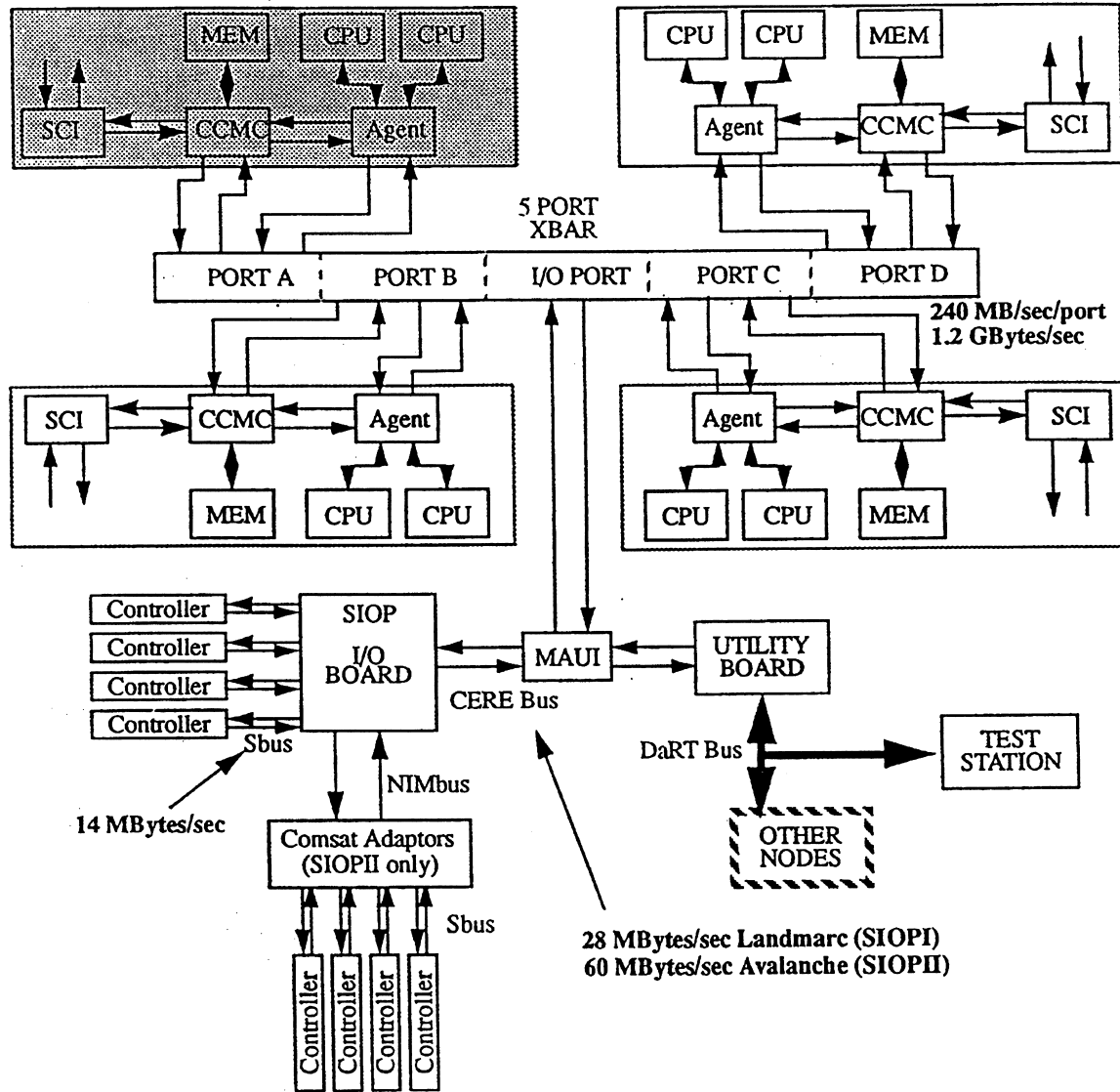
1.3.8 MAUI

- **Mpp Agent Utility and I/O.**
 - This GA is the link between the crossbar system and the I/O system.
 - It provides the interface for both the SIOP and the MU to the rest of the system.

1.3.9 SCI

- The Scalable Coherent Interface gate array provides internode memory communications.
 - There are up to 4 SCI gate arrays per node, 1 per slice.
 - It is through this high-speed link that global memory (GSM) can be accessed.

Figure 1-2
CSB block diagram



1.4 Misc. Definitions

1.4.1 MOD

- The **Meaningful Optical Display** lights are only on the XA configurations.
 - The lights provide a way for the system administrator to convey system environmental conditions (such as a load average) to an external optical indicator.

1.4.2 Front Panel

- The **front panel** is a Liquid Crystal Display that contains 4 lines x 20 characters per line of node relevant information.
 - 1 Front Panel Display per node.
 - Can be viewed without opening the cabinet.
 - Indicates system status.

1.4.3 OBP

- The **Open Boot Program** is the software interface between the test station and the node.
 - It passes the OS boot parameters as well as the boot command itself to the processors.

1.4.4 SPP-UX

- **SPP-UX** is the name of the Operating System used on the SPP.
 - It is different in many ways from ConvexOS.
 - It has been patterned after HP-UX and only differs from it where hardware forces it to be.

1.4.5 PVM

- **Parallel Virtual Machine** is a message passing system enabling several processors to run an application in parallel.
 - It was implemented via ethernet in the cluster machines.
 - In Exemplar it is implemented over the SCI ring.
 - It enables idle cpus to work the application as resources become available.

1.4.6 GSM

- **Global Shared Memory** is a vast section of physical memory accessible to every node in a complex.
 - GSM does in hardware what PVM does in software.
 - Also referred to as **CTI**
 - ◆ **Coherent Toroidal Interface**

Misc. Subsystems

2

2.1 DaRT bus

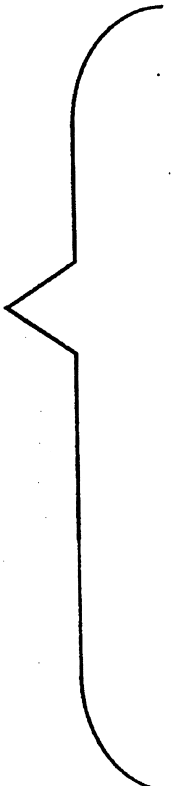
- The Diagnostic and Remote Test (DaRT) bus is an ethernet bus that is used by the Test Station to communicate to all nodes in a complex.
 - The DaRT bus provides:
 - ◆ an interface through which a node can be configured, tested and booted
 - ◆ and an avenue for the OS to log events to a file on the Test Station.
 - This interface uses an ethernet cable issued with the SPP system, which is 50 feet in length.
 - In a multinode system the DaRT bus will connect to every node (to each MU on each node), addressing each with a different ethernet address.
- Open vs. Closed Dart
 - An open dart bus refers to connecting the dart to an open local area network.
 - As this is possible it is not recommended, for obvious reasons.
 - A closed dart bus is accomplished by connecting the dart cable to lan0 on the test station and using lan1 to attach to the rest of the world.

2.1.1 DaRT and /etc/hosts

- The /etc/hosts file must contain the IP addresses of the MUs on the DaRT bus.
- The following table shows the names to be used in conjunction with the IP addresses.

Figure 2-1
Dart bus /etc/host entries

These Addresses
will be supplied
by the site.



```
130.169.128.0 mu_0000
130.169.128.1 mu_0001
130.169.128.2 mu_0002
130.169.128.3 mu_0003
130.169.128.4 mu_0004
130.169.128.5 mu_0005
130.169.128.6 mu_0006
130.169.128.7 mu_0007
130.169.128.8 mu_0008
130.169.128.9 mu_0009
130.169.128.10 mu_000a
130.169.128.11 mu_000b
130.169.128.12 mu_000c
130.169.128.13 mu_000d
130.169.128.14 mu_000e
130.169.128.15 mu_000f
```

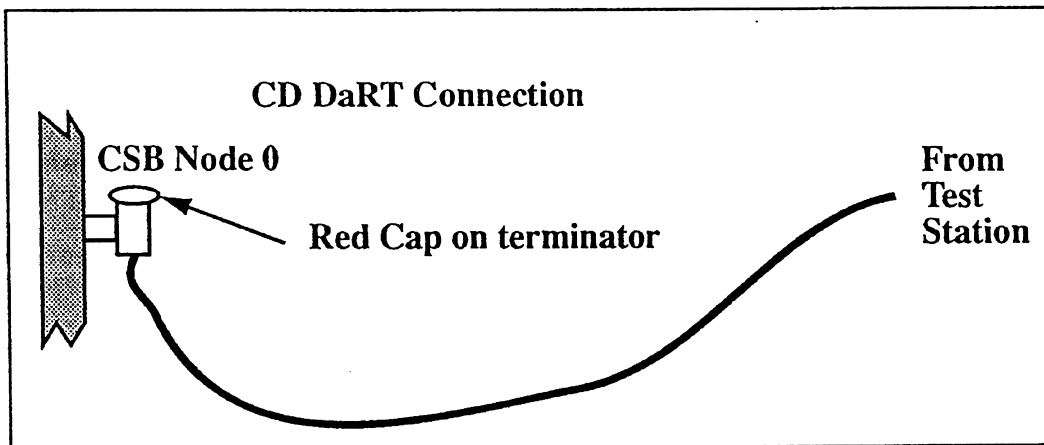
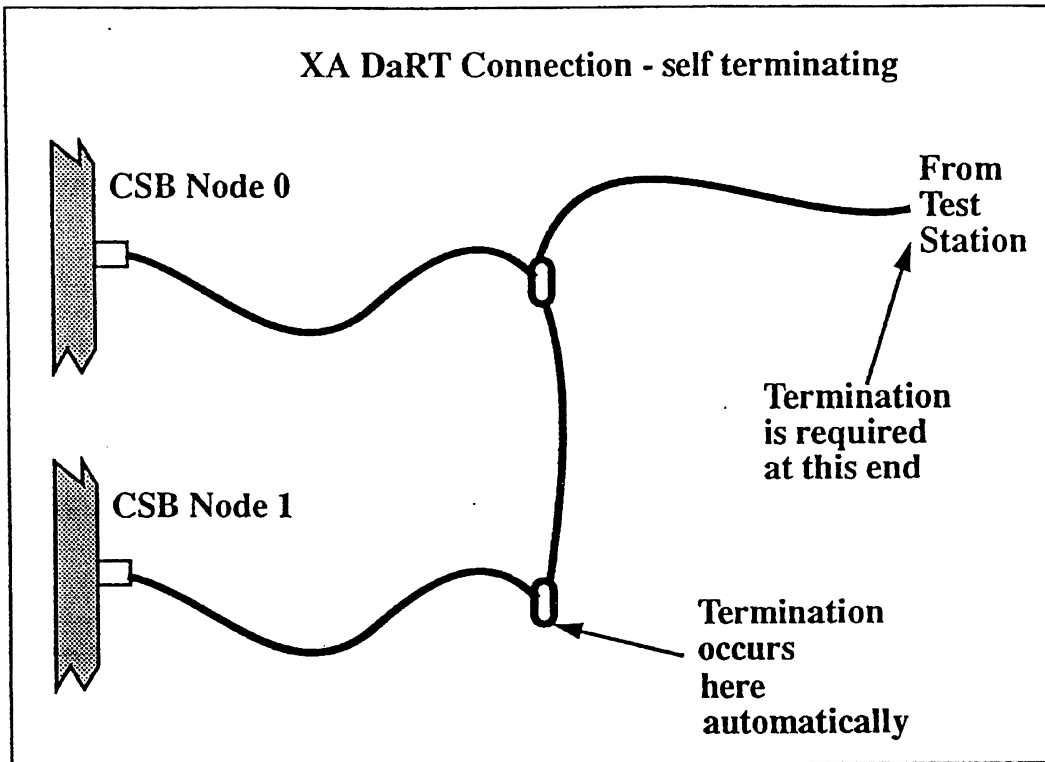
**** Note ****
The above values are the
default values supplied
at install time

**** Note 2 **** /spp/bin/node_ip_set (serial #)(node ID)(IP addr) (udp base 675)

2.1.2 DaRT Cable

- The standard DaRT bus cable sent with each node in an XA cabinet is a self-terminating cable.
 - There is no need to terminate the end of the cable that is attached to the node.
 - Dart cables on a CD unit will use terminators.
- It has been seen in the field where the self termination on the XA cable does not close properly, thus causing the ethernet connection to go *crazy*.
 - If adding a cable and this seems to be the case, simple flip the tee around and try the other side.
- The following figure shows how the DaRT bus cable is configured.

Figure 2-2
The DaRT bus cable



2.2 Front Panel Display

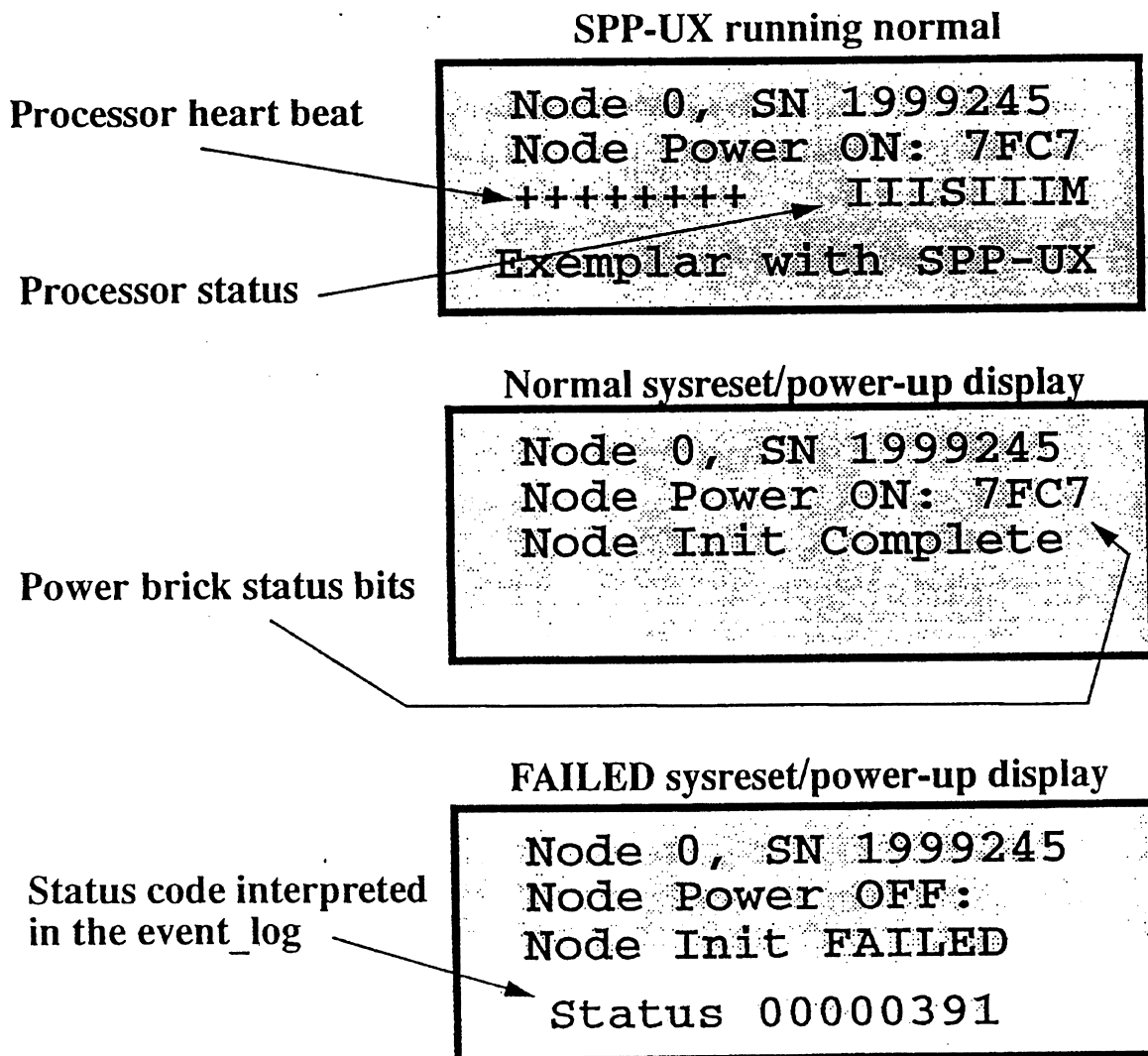
Each node has a Liquid Crystal Display which is 4 line deep and 20 characters wide.

- This LCD can be viewed from the front of the cabinet without opening any doors.
- If the MU has a failure during the power-on self test, it will report it to the front panel:
 - This may be the only source of failure information, due to the possibility that the MU may not have established communications with the Test Station at the time of failure.
- While the SPP is in OS the front panel displays the status of each processor.
- There are three different part numbers for the LCDs, depending on type of node and positioning of the node.
 - XA LCD Upper Node Cable 601-200010-201
 - XA LCD Lower Node Cable 601-200010-200
 - CD LCD Cable 500-000073-201

2.2.1 Typical Front Panel Messages

- The following figure illustrates examples of messages that can be seen on the front panel display after events such as:
 - SPP-UX booted.
 - Power on/sysreset completing successfully.
 - A MU selftest failure.

Figure 2-3
Front Panel Messages



- See Appendix G for MU status Codes.
- Description of Processor Status Letters
 - User
 - Mach Kernal
 - Emulator
 - Idle
 - Server

2.3 MOD Lights

The MOD lights (Meaningful Optical Display) are located on the front, top and back of the XA system chassis.

- They are programmable by the customer.
 - Currently the scripts `"/spp/scripts/mod/mod_all_on` and `/spp/scripts/mod/mod_all_off` are the only programs shipped with the systems.
 - One KByte of memory is available on the MU for MOD programing.

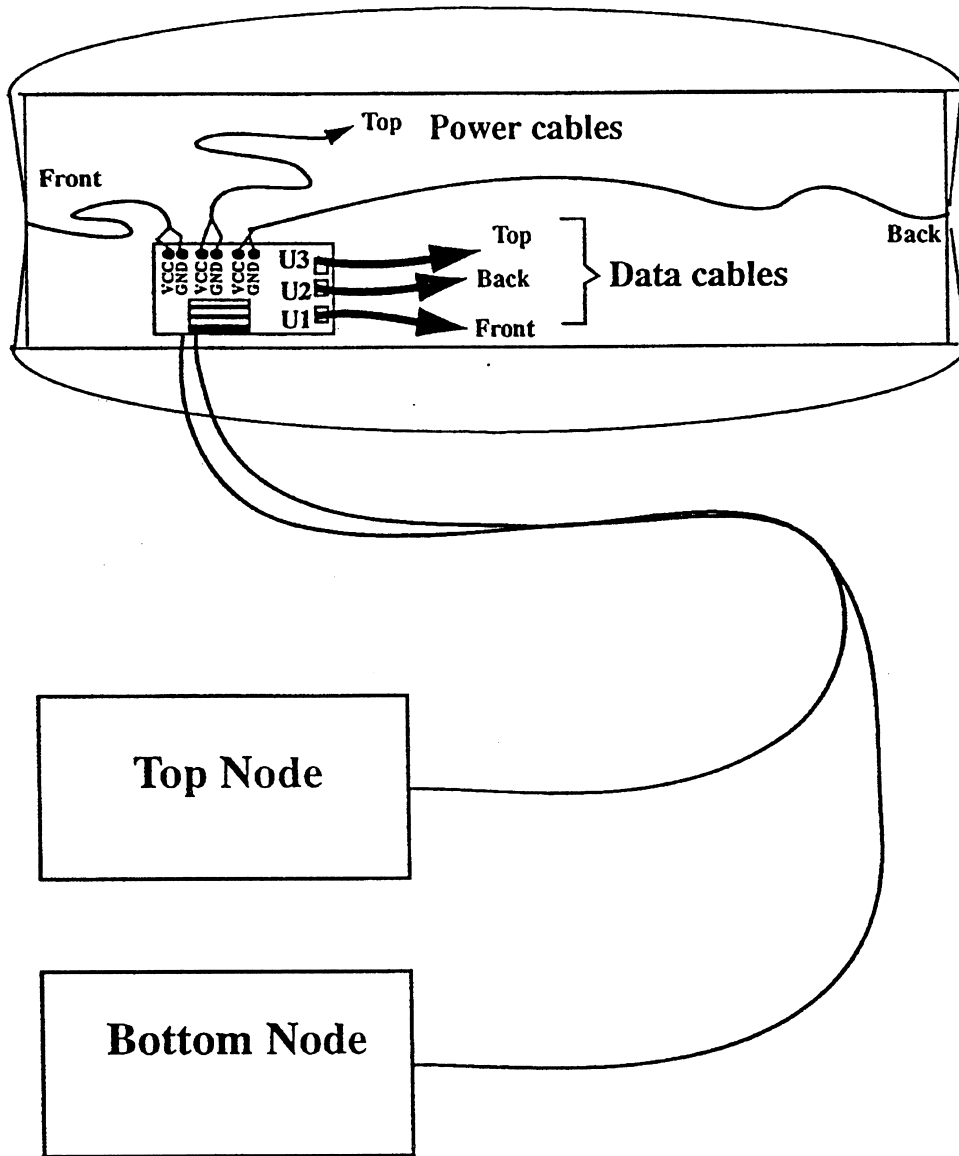
2.3.1 The MOD light controller

The MOD light controller is located in the top of the cabinet.

- One or both of the Nodes communicate with the MOD controller via a cable that connects to directly to the CSB.

2.3.2 MOD light cable routing

Figure 2-4
Mod light power and data cable routing
Top View of the cabinet

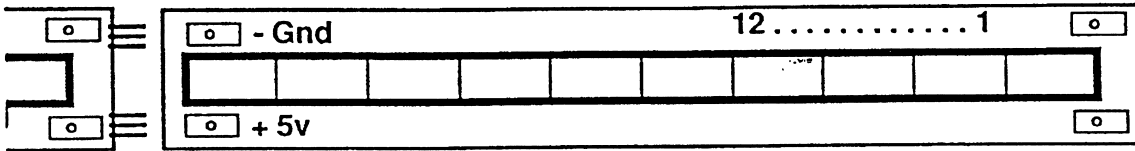


2.3.3 MOD Board Strapping

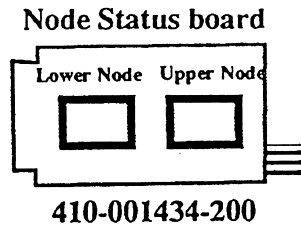
Figure 2-5
MOD Board Strapping Diagram

Total of 4 boards for each side
Plus Status board on front

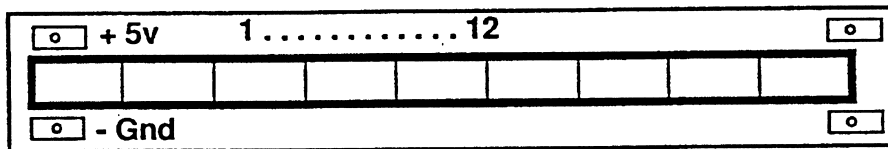
MOD LED (Front & Rear)
410-002431-200



Jumper Settings	
Upper (Board 3)	8-7
Upper Middle (Board 2)	8-7 2-1
Lower Middle (Board 1)	8-7 5-4
Lower (Board 0)	12-11 8-7 5-4 2-1



MOD LED (Top)
Total of 3 boards across the top
410-002432-200



Jumper Settings	
Front (Board 4)	1-2 4-5
Middle (Board 5)	4-5
Rear (Board 6)	1-2 11-12

2.3.4 Part Numbers

- MOD Controller - 410-002430-200
 - Upper CSB to MOD Controller cable - 601-250008-201
 - Lower CSB to MOD Controller cable - 601-250008-200
 - Side MOD LEDs cable - 601-200009-200
 - Top MOD LEDs cable 601-200008-200

Installation

3

3.1 Initial Installation

- 1) Open the cabinet.
 - 2) Open the Node chassis.
 - 3) Remove the packing material in the chassis.
 - 4) Verify that all of the boards in the node chassis are completely plugged in. (Boards have a tendency to shake loose during shipment).
 - 5) Connect CTI and TOC cables between all nodes in the complex. (See sections 3.4-3.5.)
- CD only - if the DaRT or FDDI cable is not connected to the node you must remove the power supply in order to remove the back cover of the node, thus allowing you access to the connections on the CSB.
 - The power supply is held in place with 2 screws.
 - You can move the power supply without disconnecting any cables.
 - Remember to route the DaRT bus cable and the FDDI cable through the bulkhead on the back of the CD.

- 6) Verify the power switches on the node chassis and the peripheral chassis are in the ON position.
- 7) Verify the incoming power is correct before plugging in the power cable or switching on the main breaker.
 - Install all remote cables to the power concentrator card before enabling the main breaker.
 - Plug in the main power cable and check the AC voltages at the checkpoints on the power chassis.
 - Switch on the main breaker.
- 8) Setup the test station.
 - Install the DaRT bus cable into the test station LAN port 0.
 - Powerup the test station.
 - Answer NO to the question asking you if you are ready to setup networking.
 - Once the test station is booted, login as root (password = serialbus).

Note: LAN 0 will be used for the *private* DaRT bus and LAN 1 for a public network. Most test stations will have the ethernet card for LAN 1 installed, if so the rest of step 8 should be executed.

- Use `ts.install` to setup the 2nd ethernet in test station.
 - `cd /spp/scripts/inst`
 - `run ts.install`
 - Answer yes to "use default IP addresses".
 - Answer all subsequent questions with the appropriate information. The following will be needed:
 - ◆ IP Address for the test station that will connect LAN 1 to the open network.
 - ◆ Router Address for the router to be used on the public network.
 - ◆ Name of the test station.
 - Ignore any "can not create error messages"
 - Edit `/etc/netlinkrc`, if necessary.
See "`/etc/netlinkrc`" on page 43.
- 9) Now Reboot the test station by typing `/etc/reboot -n` and cycling power on the test station when it tells you to do so.
- 10) Once the test station is rebooted login as `sppuser` (password = `spp user`)

Note: Logging in as `sppuser` will automatically start the `xwindow` environment.

- 11) Power on the Node with the keyswitch.
- 12) Wait until the front panel shows "Node init complete".
- 13) In the sspdsh window ping the node: **ping mu_0000**.
- 14) After communications with the SPP are verified, use ccmu to turn off the OBP and set any other hardware parameters as needed, then run these diagnostics in the following order:
 - Cst "System Test" , run on each node in the complex.
 - On a multinode system run verify_cables.
 - Ccmu pull (or do_reset).
 - Sppring -s 1000-1050,2000-2020,2120,3000-3020,3120,3220,4000-4010 and 5000-5010 (also run class 6,7,8 and 9 if multinode).
 - Do_reset.
 - Io_tc (this will test the SIOPs).
- 15) Ccmu pull.
- 16) Configure OBP parameters as needed.
- 17) Do_reset (always before booting).
- 18) Type **boot** in the System Console window at the OPB prompt - "[0:1] ok".

3.2 /etc/netlinkrc

- With older versions of diagnostic software it was necessary to do the following if the test station had 2 LAN's:
 - Look for the following lines in the netlinkrc file:

```
# IF THE TEST STATION HAS A SECOND NETWORK CONNECTED, AND IT IS  
CONNECTED  
# TO AN ACTIVE NETWORK, REMOVE THE COMMENTS BEFORE THE NEXT 12 LINES
```

- Do the above, then reboot the test station.
- Current revisions of test station diagnostic software will already have the comments taken out of the 12 lines mentioned above in the netlinkrc file.

If only one lan is used, then comments will have to be added.

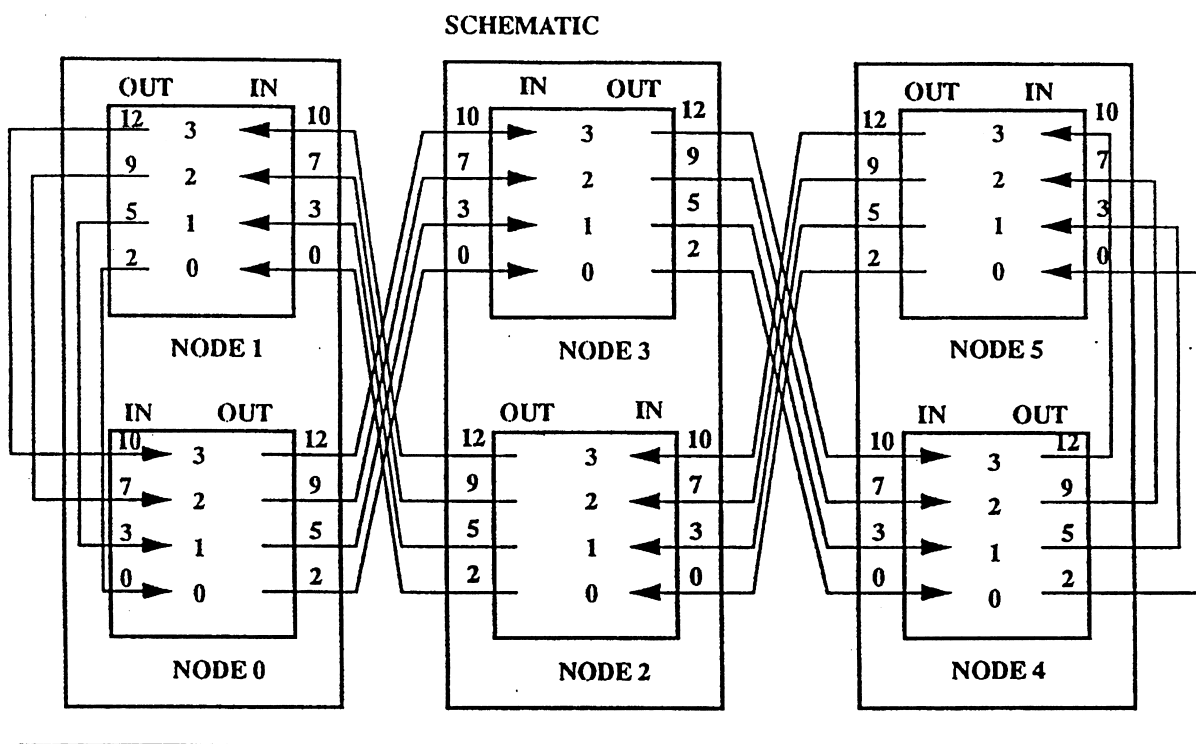
3.3 Autoconfig network on boot

The first time SPP-OS is booted you will be prompted for network information. This information will be used to auto configure the network files on the system.

- The information you will need to supply is:
 - Your system name (host name)
 - Your internet protocol (IP) address.
 - The local router IP address, if there is one.
 - Your time zone.
- If you do not have this information, you may set these parameters when they have been obtained by using `'/etc/set_parms'`.
- If you wish to be prompted by the autoconfig network screen like the initial boot sequence, remove the following files and reboot:
 - `/etc/src.csh`
 - `/etc/src.sh`

3.4 SCI RING Connection

Figure 3-1
SCI XA Cabling Diagram



TOP VIEW

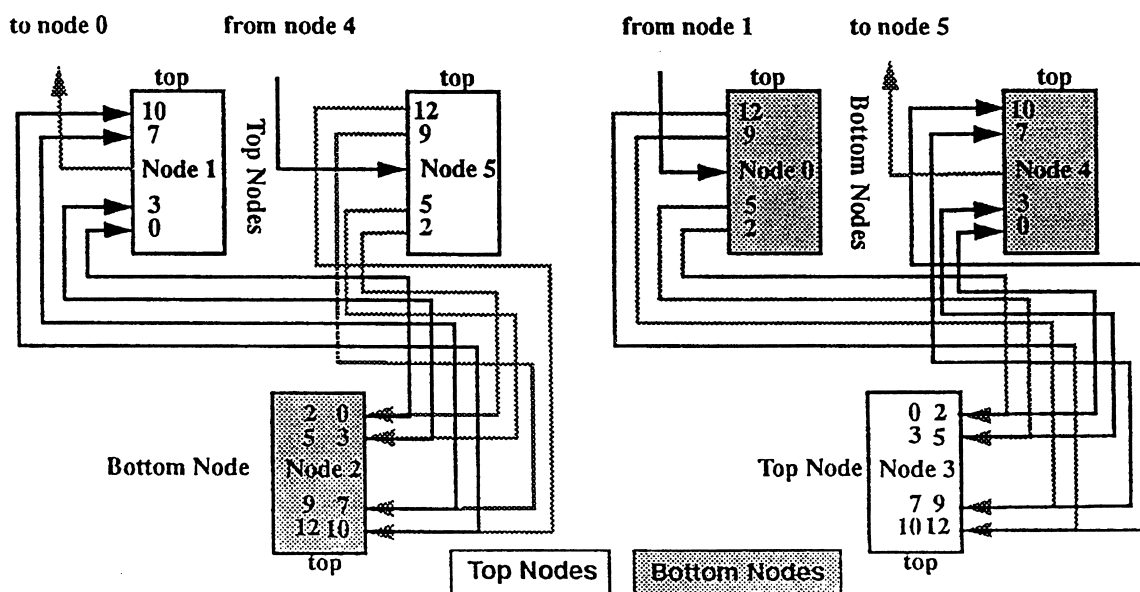


Figure 3-2
Node XA SCI Cabling Diagram

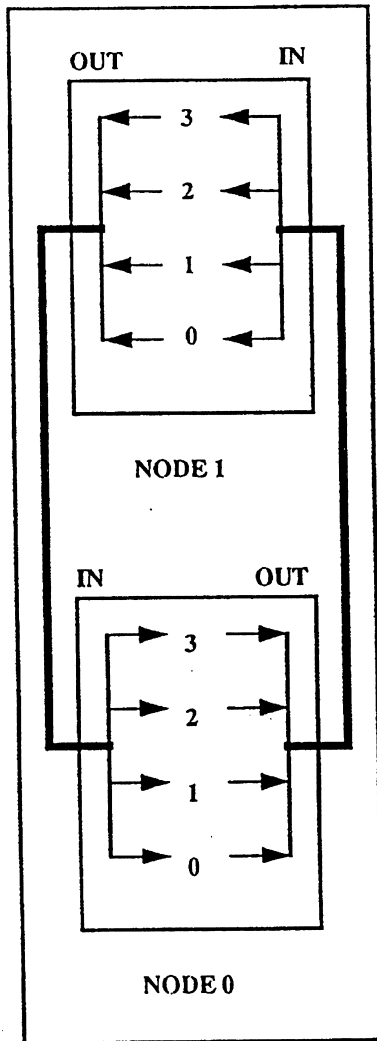
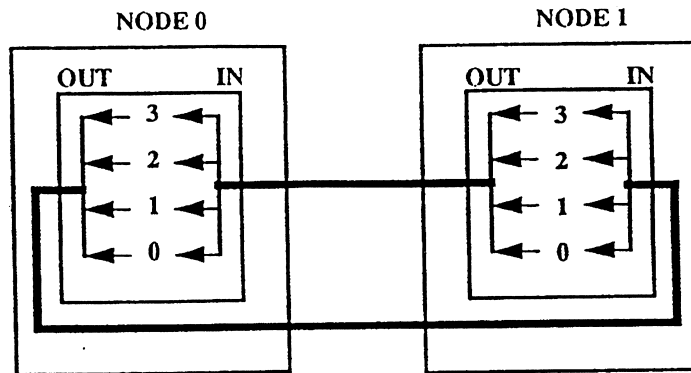


Figure 3-3
SCI CD Cabling Diagram



3.4.1 SCI diagnostics

- `verify_rings`
- `verify_cables`
- `sppring -c 6,7,8,9`

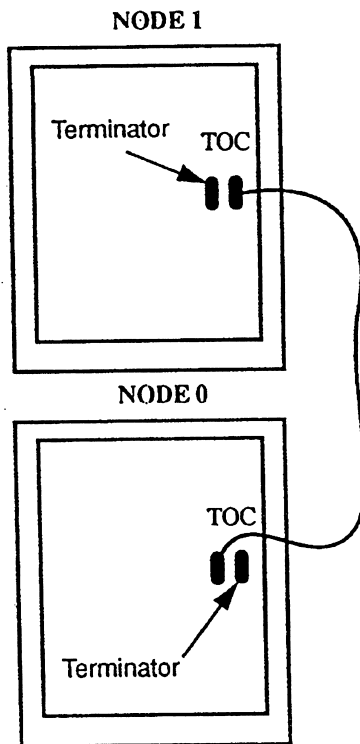
3.4.2 Multi-cabinet connection & Power sequencing

- See wiring diagram 550-000053-300 pages 1,2&3

3.5 TOC cable Connections

- The Time Of Century cable is connected in a "Daisy Chain" fashion.
- The convention of "IN" on the left connector and "OUT" from the right connector is suggested to be followed.

Figure 3-4
TOC cable connection diagram



3.6 Configurations

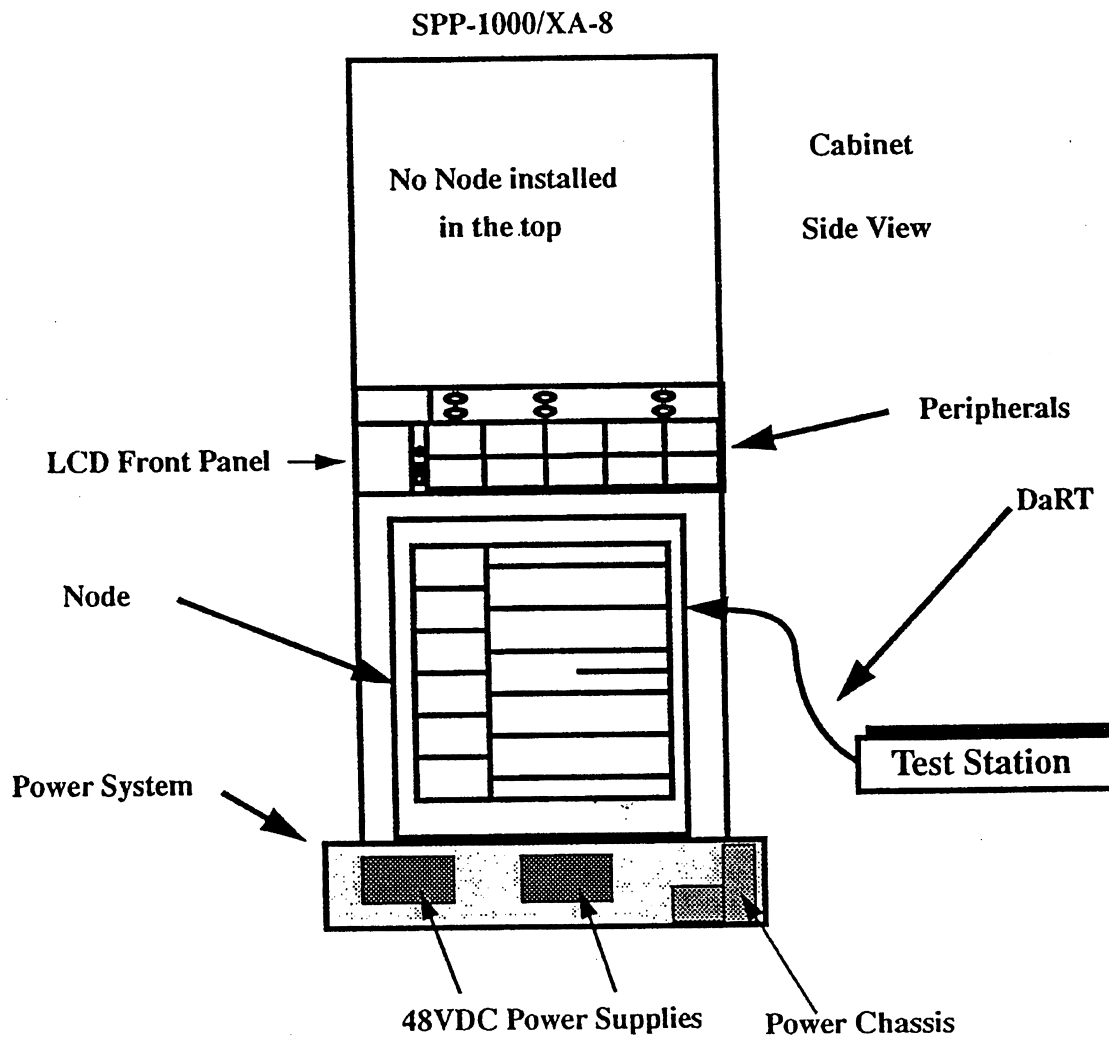
This section contains different configurations of the Exemplar series systems. A complex can range in size from 1 to 16 nodes, contained within 1 to 8 cabinets.

- XA Single Node
- XA 2 Node
- CD Single Node
- XA Single Node Multi-IO
- CD Multi-IO
- CD Maximum Configuration
- XA Maximum Configuration
- XA Standard 6 Node Configuration

3.6.1 XA Single Node

- XA - eXtended Architecture
- A single node XA weighs approx. 500 lbs.

Figure 3-5
XA - Single Node Configuration



3.6.3 CD Single Node

Figure 3-7
CD Single Node configuration

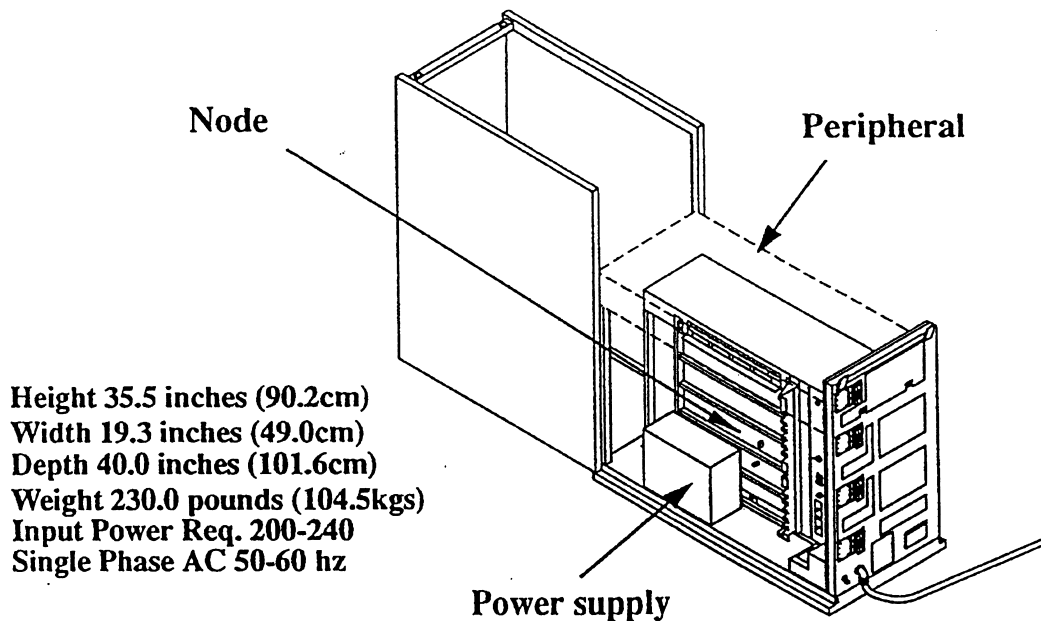
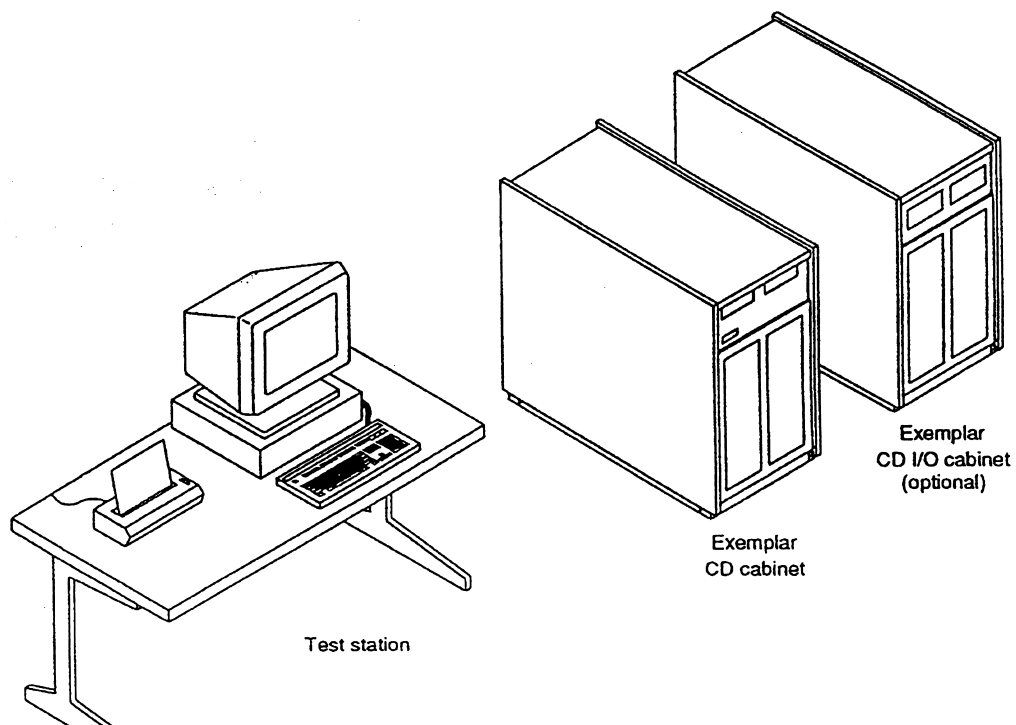


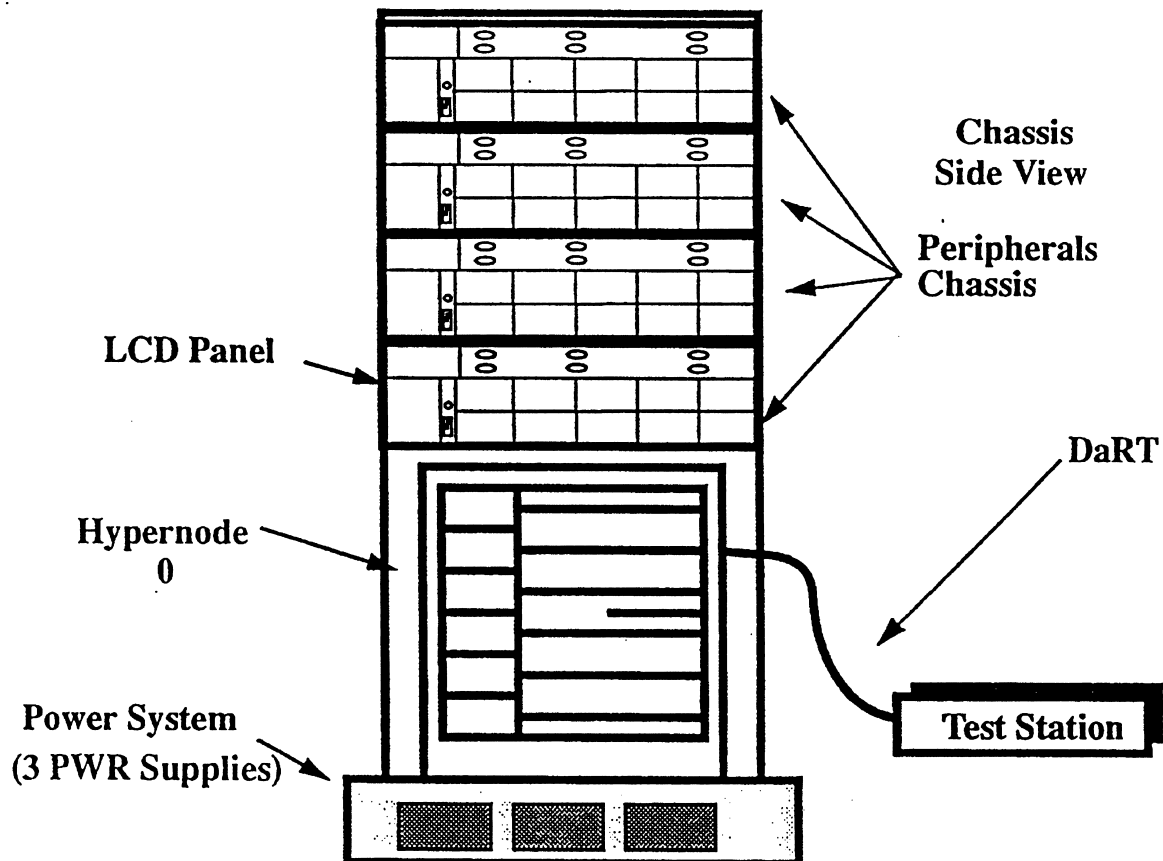
Figure 3-8
CD W/Test Station and IO chassis



3.6.4 XA Single Node Multi-I/O

- 4 IO chassis.
- 3490s can be installed in upper 3 locations.
- 80 disk drives max, with 4 IO chassis.
 - Bottom chassis must be I/O.

Figure 3-9
XA Single Node Multi-I/O configuration



3.6.5 CD Multi-IO

- The I/O Chassis on the CD can be a differential or single ended SCSI.
 - The current configurations will contain the new Differential I/O Chassis.
- The I/O Expansion cabinet will contain the High Density (XA type) IO Chassis or Differential Chassis.
 - IE: it MUST be driven by a differential SCSI controller.

Figure 3-10
CD - Multi-IO

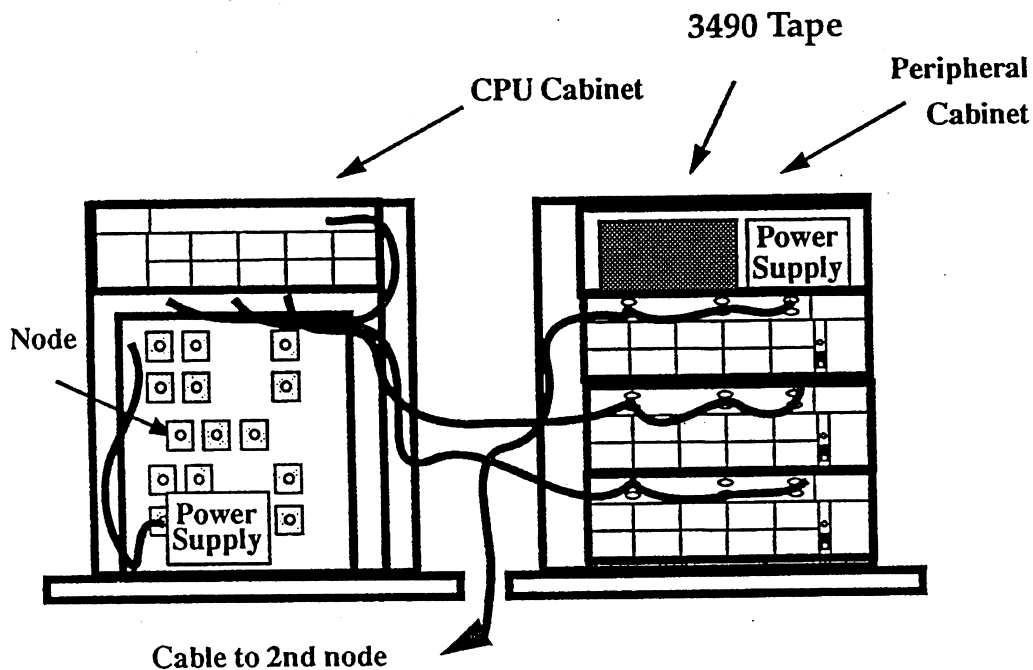
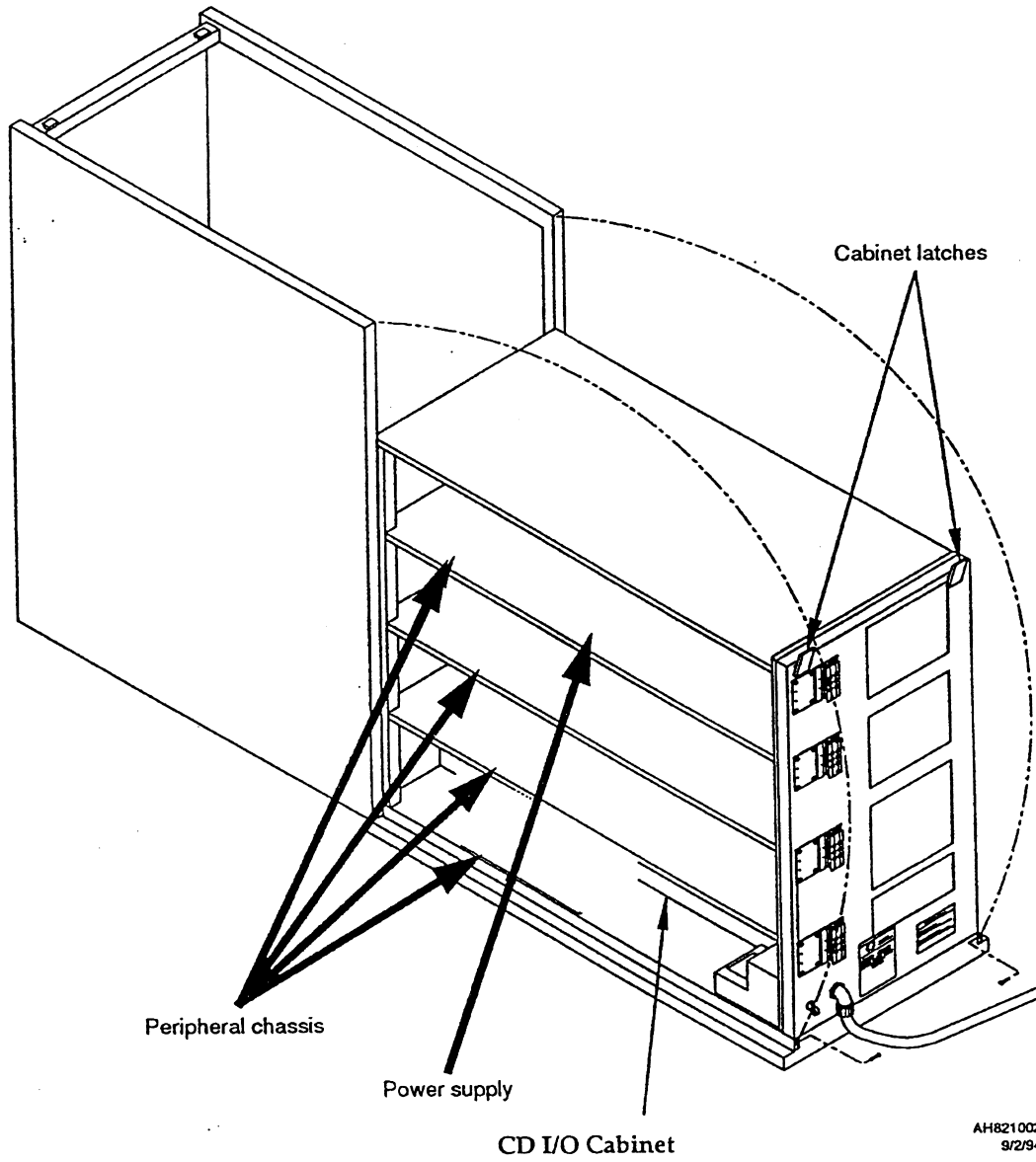


Figure 3-11
CD - I/O Chassis and cover



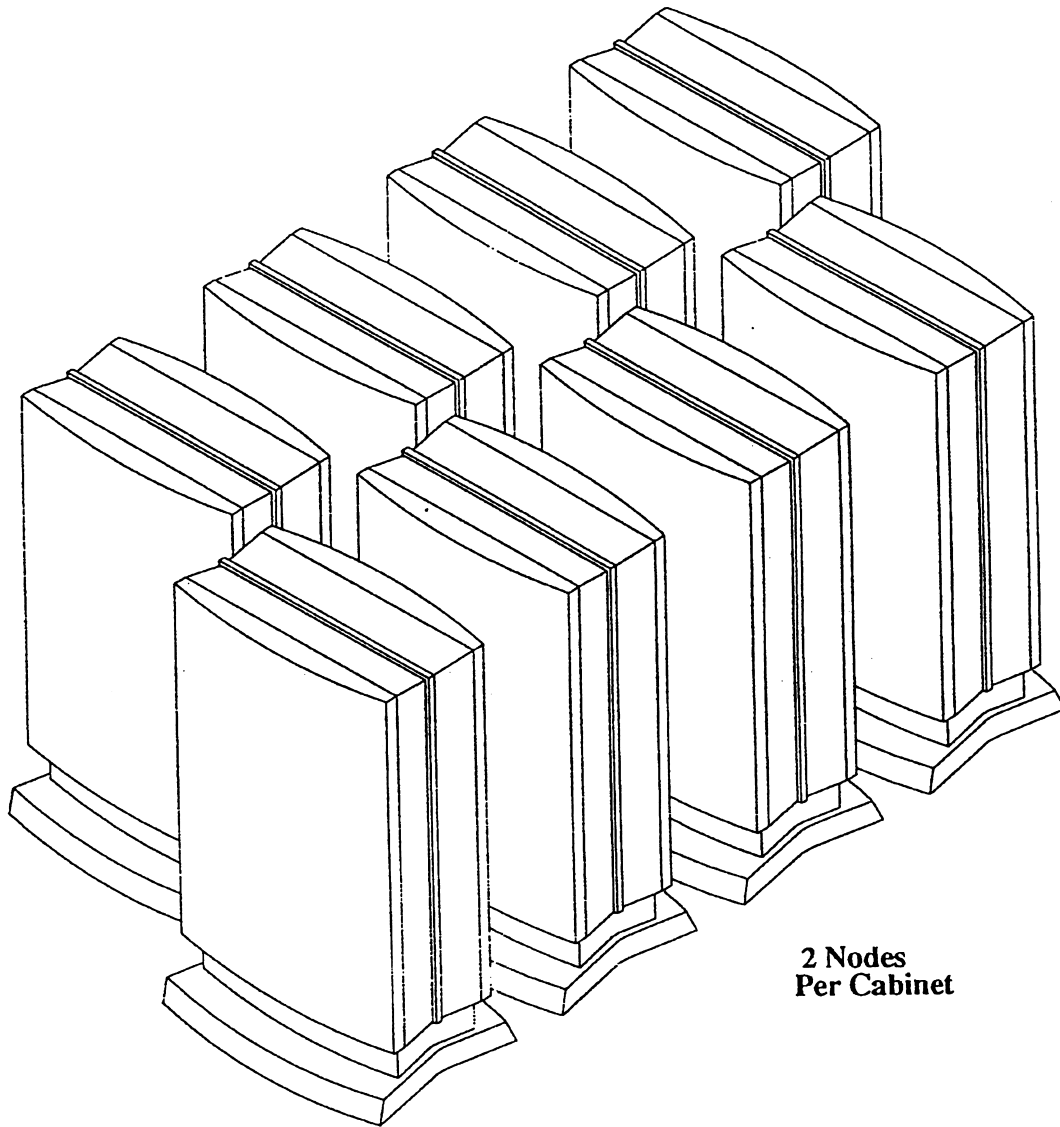
3.6.6 CD - Max Configuration

- 2 Nodes.
- 16 CPUs.
- 2 GIG memory (512M MTVs).

3.6.7 XA - Max Configuration

- 8 CPUs Per Chassis (Node).
- 16 CPUs Per Cabinet.
- 128 CPUs Maximum configuration of a Complex (16 Nodes).
- 32 Gigabytes memory w/ 512M MTVs.
- 64 Gigabytes w/ 1Gig MTV (N/A Yet).

Figure 3-12
16 Node Configuration
Maximum Configuration



AH500021
10/15/93

3.6.8 Standard 6 Node Configuration

Figure 3-13
XA 6 Node Configuration

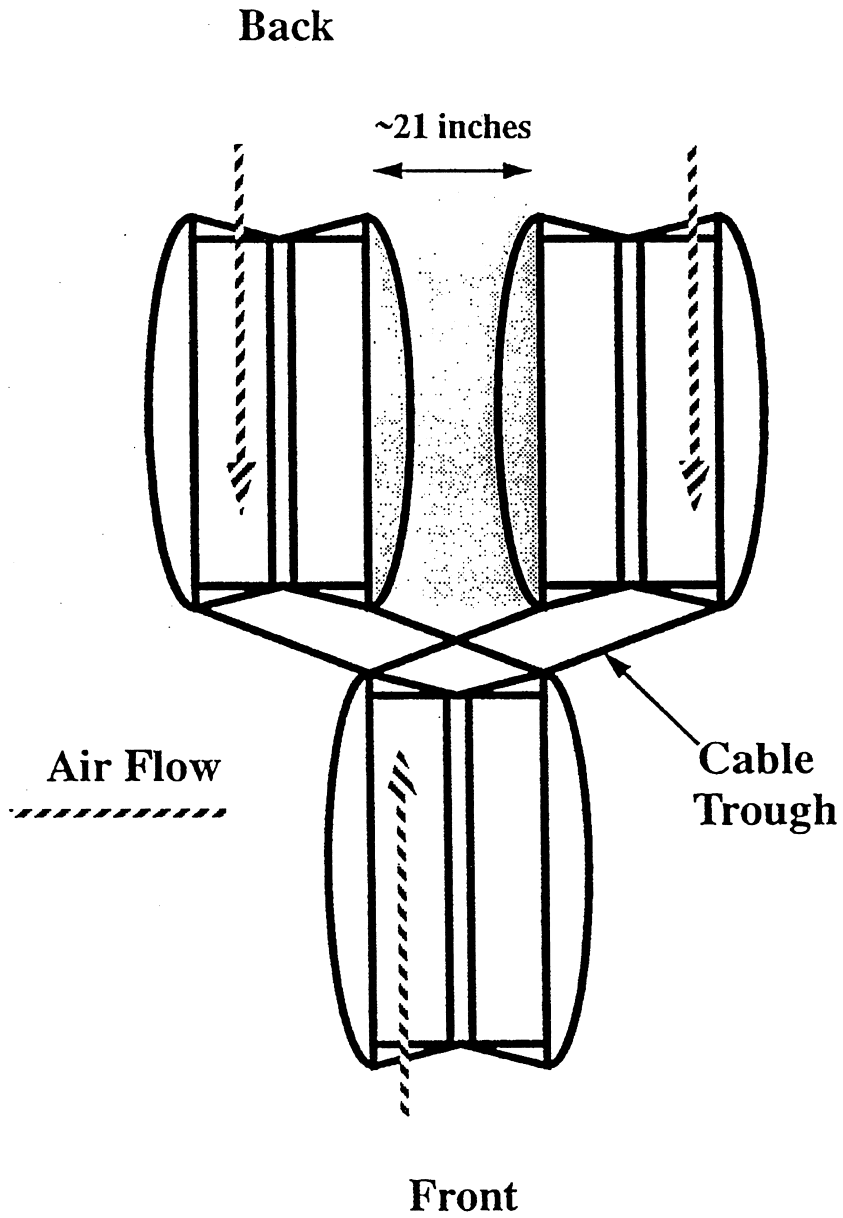
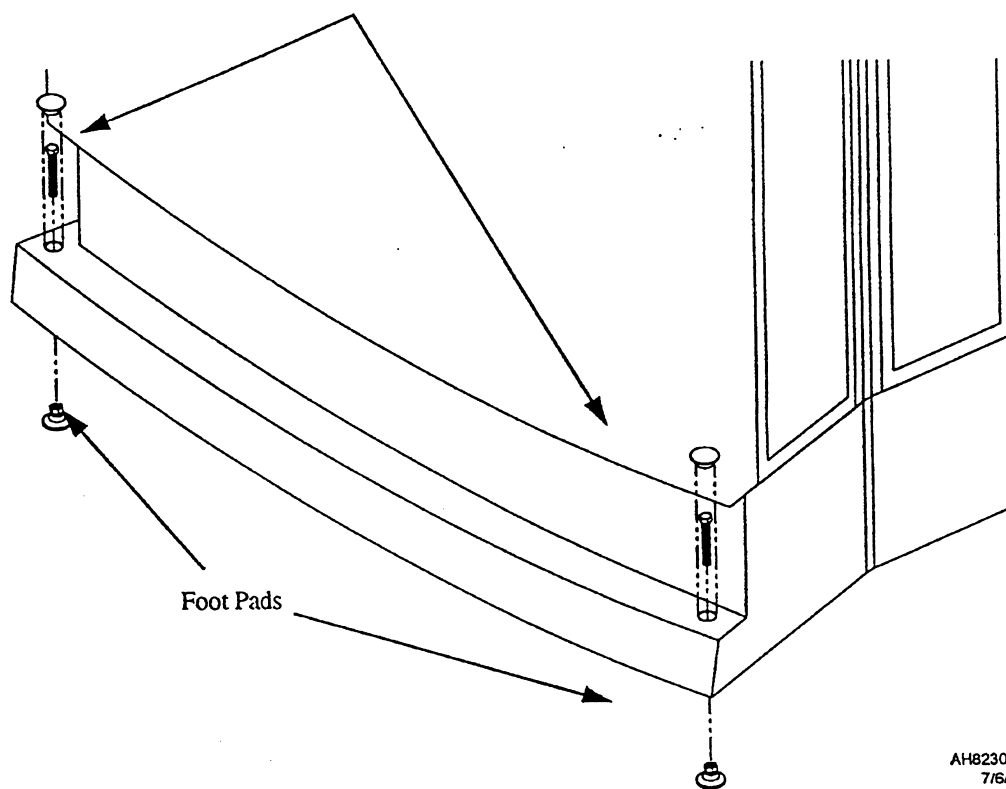


Figure 3-14
XA Leveling Feet

The XA Cabinet leveling feet are installed from the top.



The Power System

4

4.1 Introduction

The power system on the Exemplar series consists of:

- Power Brick Boards
- Power Bricks
- Power Supplies
- Power Chassis
- Power Concentrator Card

4.2 Power Sequencing (ON/OFF)

It is very important to follow the proper power up/down sequencing when applying and removing power on the SPP. Failure to do so can damage the SPP.

- Power on sequence:
 - MUon
 - main 48 on

OR

 - keyswitch on (if the SPP was powered down with the keyswitch and/or both MU and Main power switches are already on)

- Power off sequence:
 - pce_util -p off
 - main 48 off
 - MU off

OR

 - pce_util -p off
 - keyswitch off

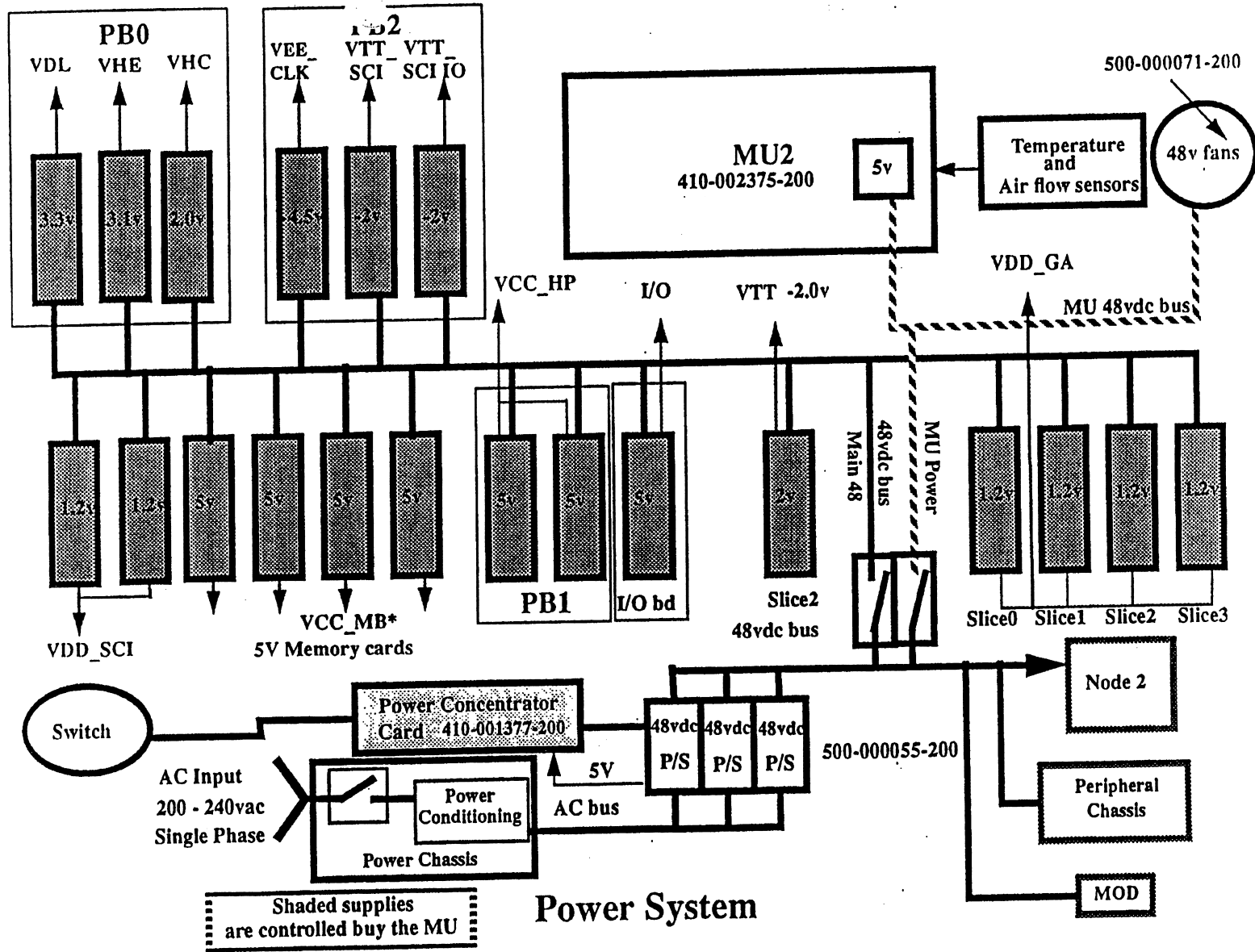


Figure 4-1
Power Subsystem Diagram

4.3 Power Brick Descriptions

4.3.1 Power Board 1

- Power Board 1 contains the power brick that supplies VCC_HP (+5vdc) to the T-Chip Modules (main power).
- Part Number 410-001379-200.

4.3.2 Power Board 0

- Power Board 0 contains the power brick that supplies:
 - VHC (+2vdc) HECL (GTL-like) VCC supply (ECLinPS logic, and PLL).
 - VHE (-3.1vdc) VEE-HP Power for the T-Chips.
 - VDL (+3.3vdc) T-Chip modules (I/O power) also sent to MU, Landmarc and Fujitsu gate arrays.
- Part Number 410-001378-200.

4.3.3 Power Board 2

- Power Board 2 contains the power brick that supplies:
 - VPSMU (+48vdc) Main power for the MU (VCCMU) and SCI bricks (VTTGA, VTTSCI and VDLSCI).
 - VTTSCI (-2vdc) SCI Gate Arrays (TTL I/O).
 - VTTSCI (-2vdc) SCI incoming bus terminators (referenced to GNDSCI).

-
- VEE (-4.5vdc) HECL (GTL-like) VEE supply (ECLinPS logic, and PLL).
 - GNDSCI (~0) SCI incoming ground.
- Part Number 410-002380-200.

4.3.4 200 Series Power bricks

- 48v to 1.2v 40 amp converter:
 - Driver - 200-001049-200
 - Booster - 200-001049-201
- 48v to 2v 40 amp converter:
 - 200-001047-202
- 48v to 3.3v 40 amp converter:
 - 200-001047-206
- 48v to 5v 40 amp converter:
 - 200-001047-209
 - 200-001047-210
- 48v to 12v 16 amp converter:
 - Driver - 200-001047-210
 - Booster - 200-001047-211
- 48v to 4.4v 40 amp converter:
 - 200-001045-202

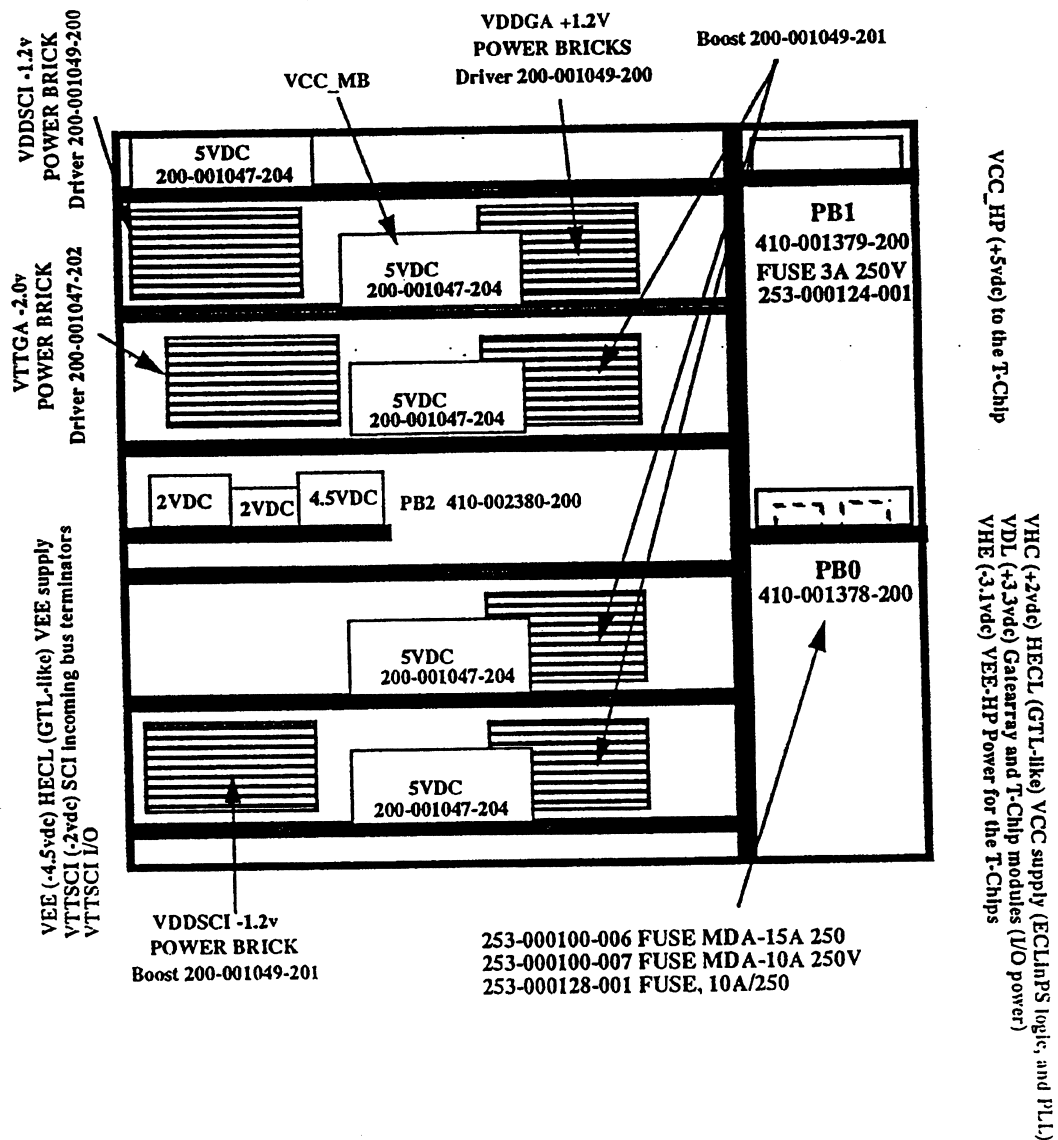
4.3.5 APB Avalon Power Board

- The APB takes the place of PB0, PB1 and PB2:
 - **VDD_GA +1.2V** Fujitsu Gate Arrays (array power) (4 large bricks on ASB).
 - **VTT -2V** Fujitsu Gate Arrays (ECL I/O power + Terminators),(1 large brick on ASB).
 - **VDD_SCI +1.2V** SCI Gate Arrays core voltage. (2 large bricks on ASB).
 - **VTT_SCI -2V** SCI incoming bus terminators (referenced to GNDSCI), (Avalon Power Board).
 - **GNDSCI ~0** SCI incoming bus ground.
 - **VCC_HP +5V** HP PA-RISC T-chip Module & SRAM power.
 - **VDL +3.3V** HP PA-RISC T-chip Modules (I/O power) Also LANDMARC and Fujitsu gate arrays.
 - **VHC +2V** HECL (GTL-like) VCC supply (ECLinPS logic, and PLL).
 - **VEE_CLK -4.5V** ECLinPS Logic
 - **VTT_SIO -2V** For SCI node termination referenced to ground.
 - **VDH_HP +4.4V** For T' Modules, main power. Actually implemented using +5V bricks.
 - **V9NDRT -9V** Ethernet, DART bus (small DC-DC on ASB fed with VCCMU, referenced to ethernet ground).
 - **GNDDRT ~0** Ethernet, DART bus ground.
 - **GND 0** Logic Ground/Chassis Ground.

4.4 Power brick locations

- Brick locations are the same for both the CSB and ASB.
 - The PB0 and PB1 power brick boards do not exist on the ASB.
 - APB replaces the PB2 on the SPP-1200.

Figure 4-2
Power Brick Locations



4.5 Power Brick Torque

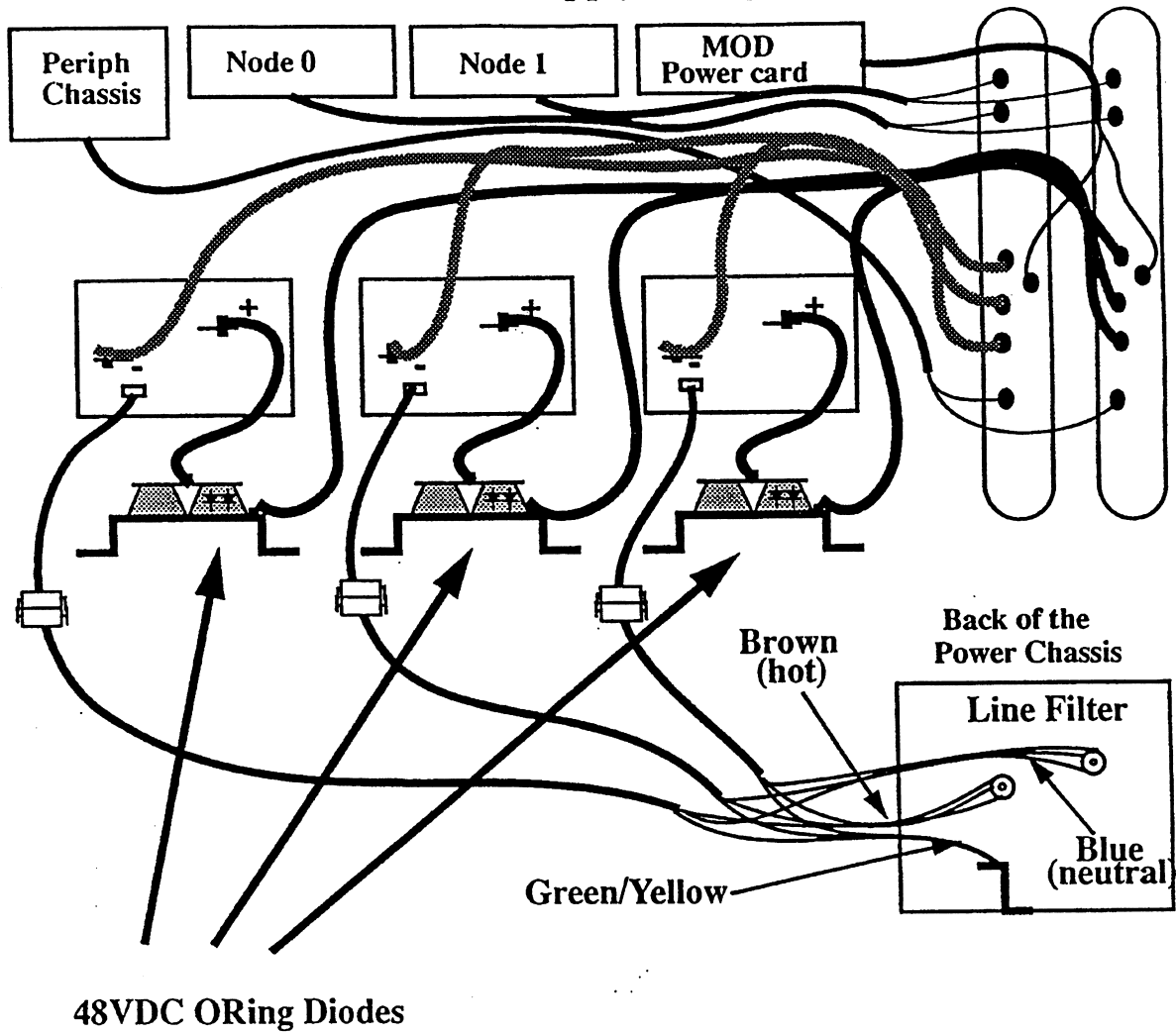
- All power bricks should be torqued to 5 in/lbs.
 - Over torquing can cause damage to the bricks.
 - It is very easy to break off the lug bolt.

4.6 48V Power Supply Adjustment

- The 48 volt power supplies are adjustable. The correct voltage should be between 48.5 and 49 volts.
- The adjustment screw is located on the 48 volt supply just above the 25 pin D connector, on the terminal end of the supply.

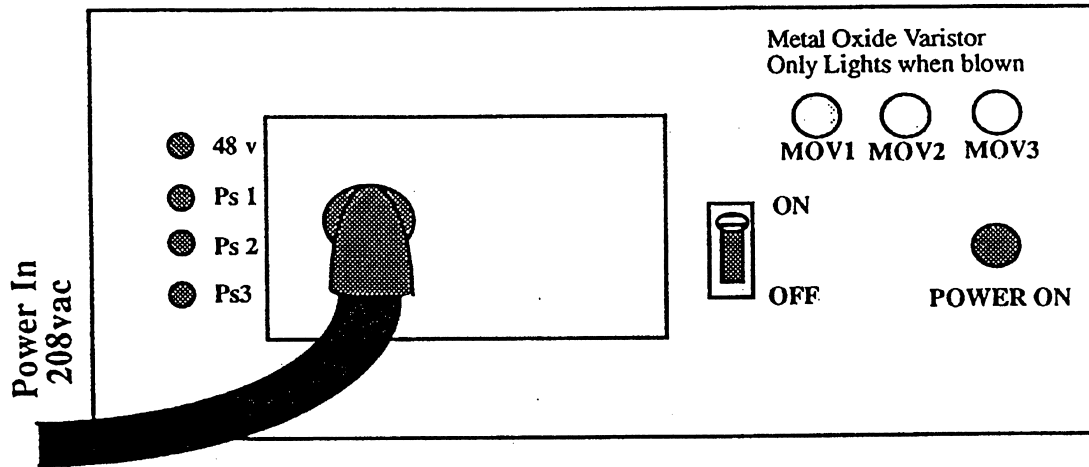
4.7 Power Chassis Wiring

Figure 4-3
Power Supply Wiring



4.8 Power Chassis Front Panel

Figure 4-4
Power Chassis Front Panel

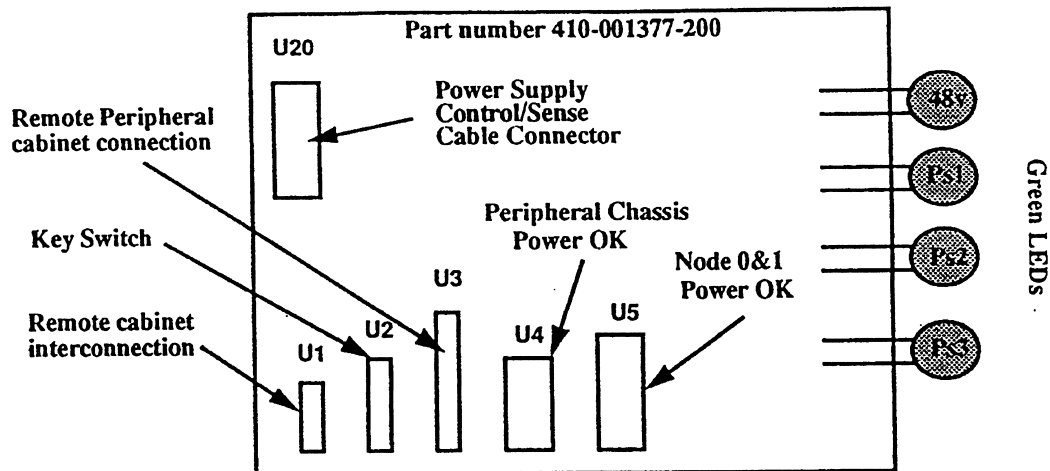


(Rear View)

- Input AC: 200VAC - 240VAC

4.9 Power Chassis Concentrator Card

Figure 4-5
Power Concentrator Card



- **Remote peripheral cabinet connection:** Connect to external peripheral cabinets for remote power up.
- **Key switch:** Cables to toggle switch on front of the SPP cabinet (located in same area as the LCDs). Controls power on for *both* nodes.
- **Remote cabinet interconnection:** Connect to other SPP power concentrators for remote power up.
 - U1 on master will connect to U2 on slave.
- **Power supply control/sense:** Cables to local power supplies for control/ sense.
- **Peripheral chassis power OK:** Cables to local peripheral chassis to monitor power OK.
- **Node 0&1 power OK:** Monitors power on for both the nodes.

CSB/ASB Level Functional Units **5**

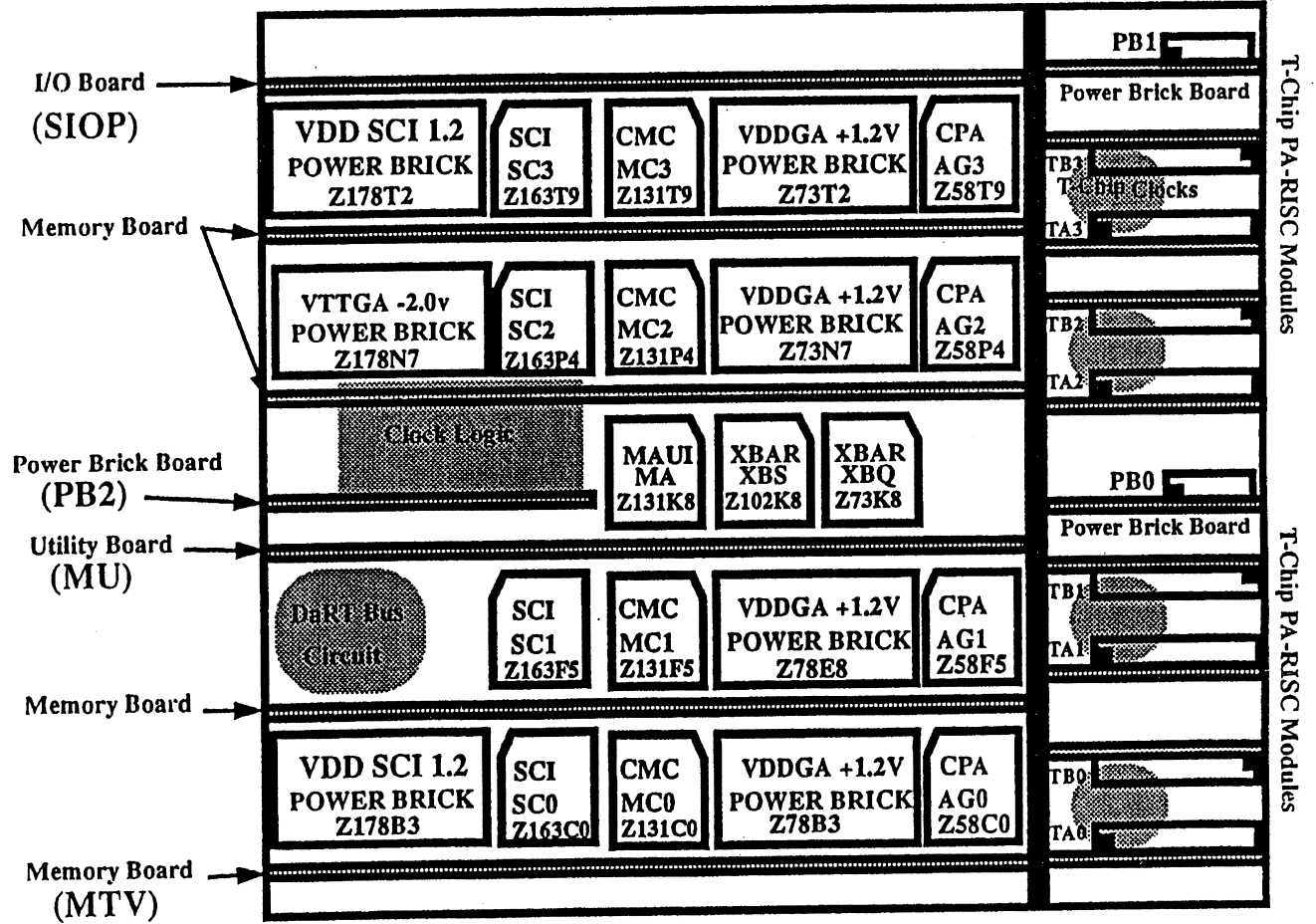
5.1 Introduction

This chapter will cover in more detail the system boards and each functional block contained within them. Both gate arrays and the daughter boards that attach to the CSB/ASB will be covered in this chapter.

5.1.1 CSB4 Physical Layout

- The CSB has both logic boards and gate arrays directly attached it.
- Figure 5-1 on page 74 and Figure 5-2 on page 75 show the physical layout of the CSB.

Camelot System Board CSB



(Z side or Bottom View)

Figure 5-1
CSB Layout Z-Side

Figure 5-2
CSB U-Side Layout

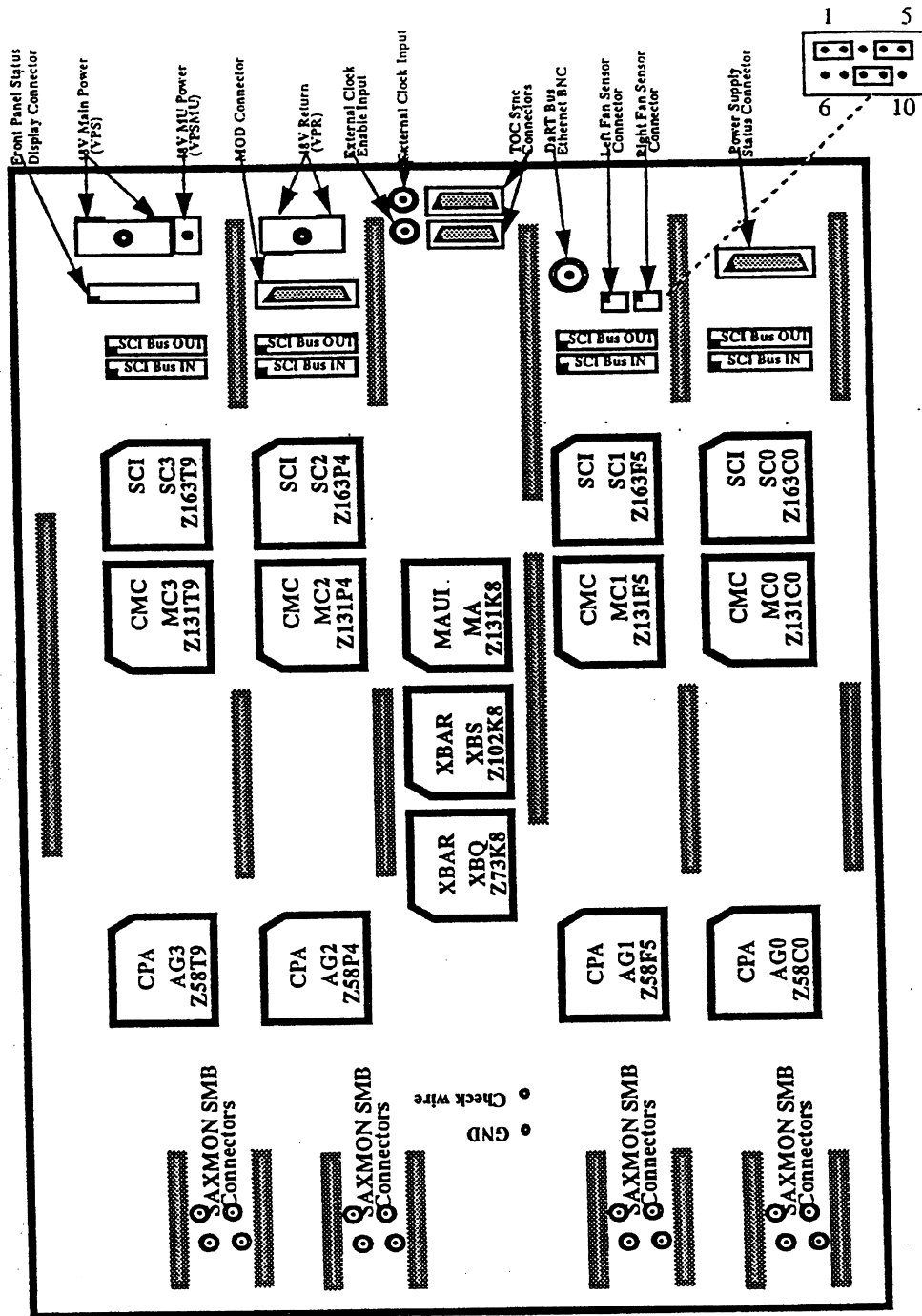
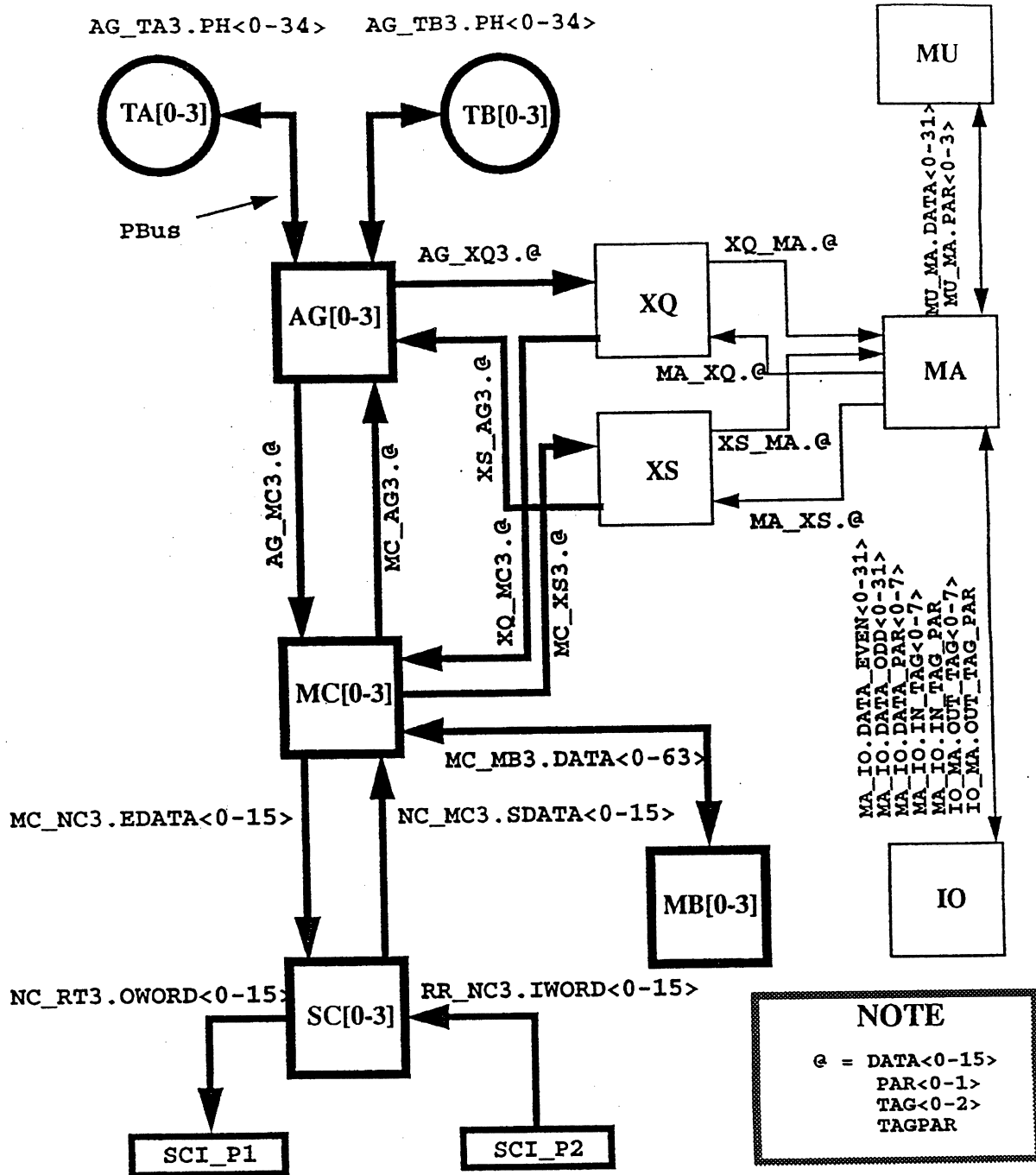
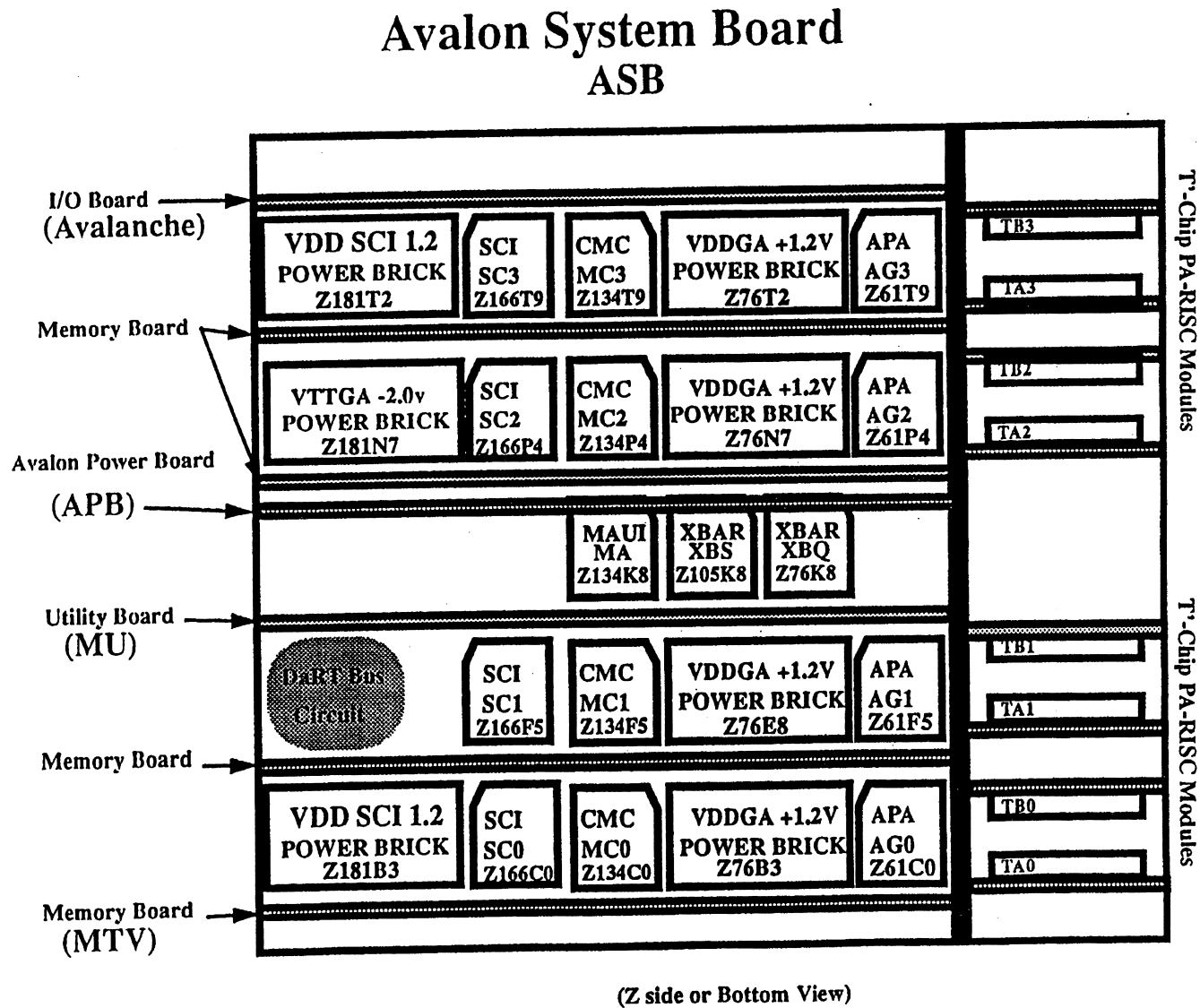


Figure 5-3
CSB Data Paths



5.1.2 ASB Physical Layout

Figure 5-4
ASB Physical Layout



- Must have APAs and XBAR2 GAs

Figure 5-5
ASB U-SIDE

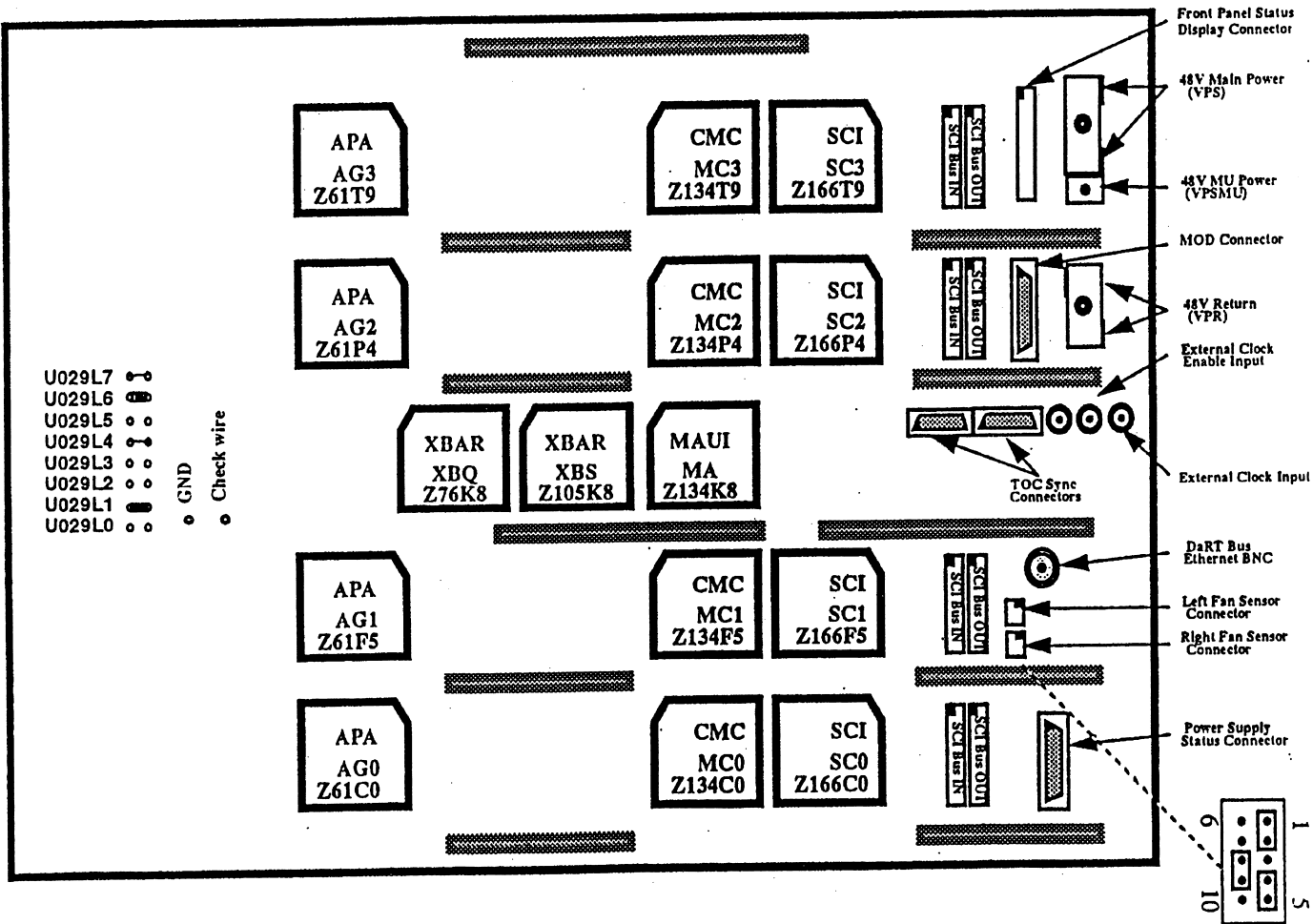
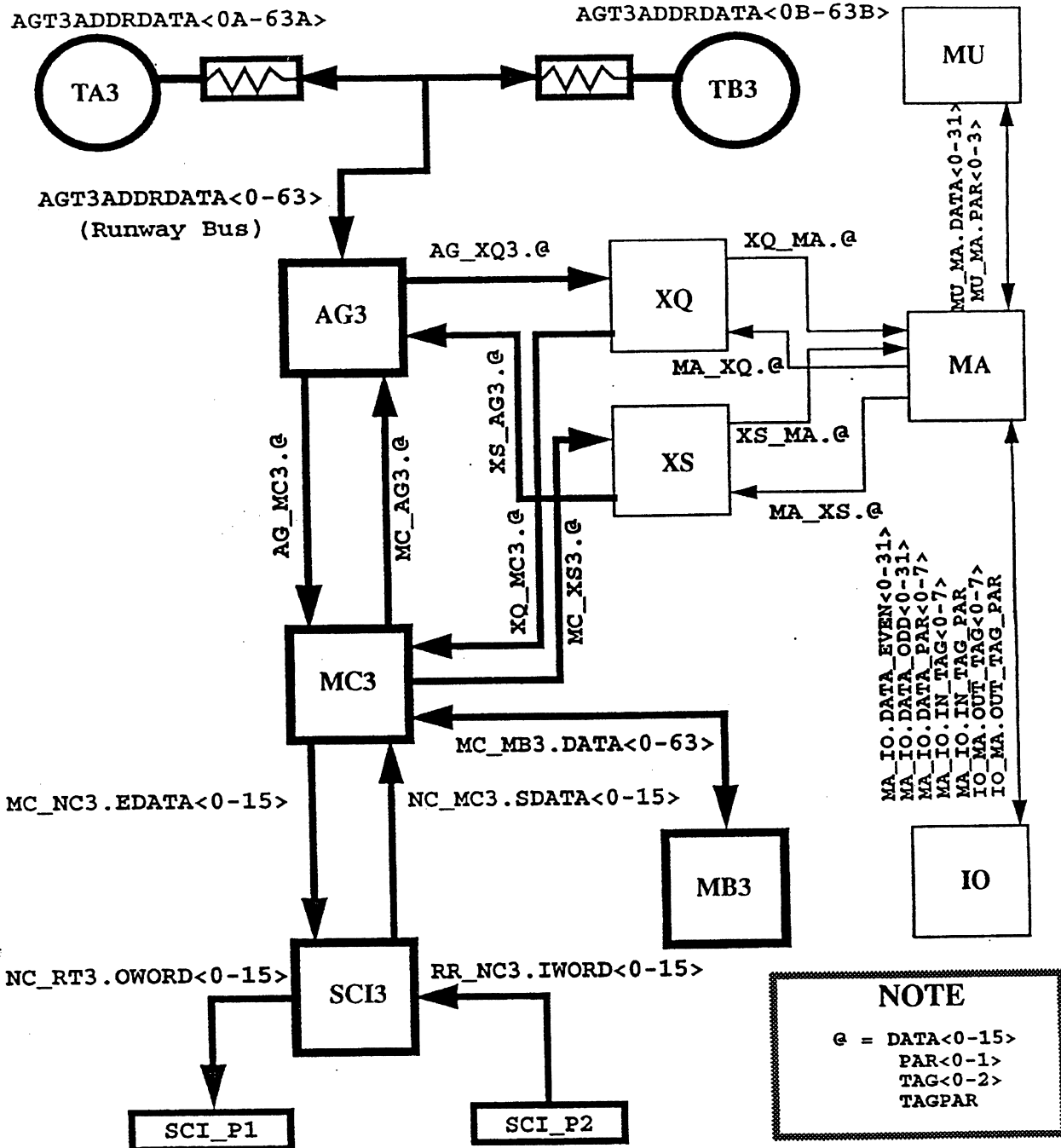


Figure 5-6
ASB Data Paths



* Signal names use slice 3

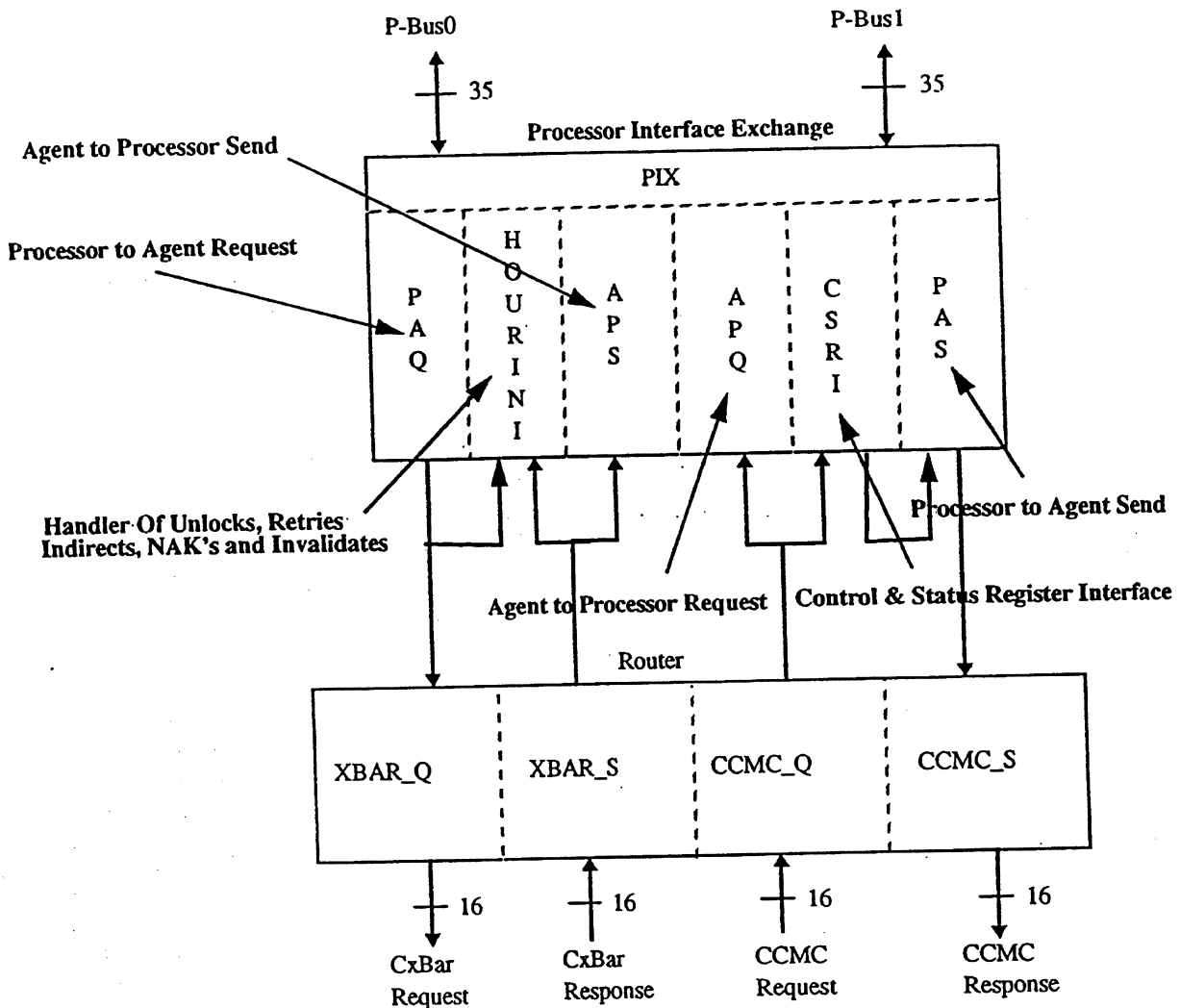
5.2 Agent Gate Array

The Agent gate array interfaces the processors in a slice to the rest of the system.

- It communicates directly with the memory control gate array and the 2 crossbar gate arrays.
- The silk screen on the CSB designates it as (AG).
- There is one Agent per slice, for a maximum of 4 Agents per node.
 - *Two processors share an agent.*
- Internally, the Agent provides 3 main functions:
 - Translates P-Bus transactions into Xbar messages and visa verse.
 - It helps maintain and manage intra-node cache coherency.
 - Provides message routing between the CCMC and the Xbar arrays.
- Figure 5-7 on page 81 shows a block diagram of the Camelot Processor Agent gate array (CPA).

5.2.1 Agent Gate Array Block Diagram

Figure 5-7
Agent Gate Array Block Diagram



****Note**** *The request-in port allows the CCMC to send processor invalidates, read_indirects, interrupts, and MAUI CSR requests from the CCMC, or route through requests from the local node. The response-out port allows the responses to be sent back to the issuing processor or CCMC.*

5.2.2 Agent Hard Errors

Table 5-1 Agent Internal Errors

cpa/hu_db_rpe	Hourini DB Ram Read Parity Error
cpa/hu_db_wpe	Hourini DB Ram Write Parity Error
cpa/hu_reply_ram_rpe	Hourini Reply Ram Read Parity Error
cpa/hu_reply_ram_wpe	Hourini Reply Ram Write Parity Error
cpa/paq_bdt_rpe	PAQ BDT Ram Read Parity Error
cpa/paq_bdt_rpe_csr	PAQ BDT Ram CSR Read Parity Error
cpa/paq_bdt_wpe	PAQ BDT Ram Write Parity Error
cpa/paq_dio_wrt_size_err	PAQ invalid size on DIO write
cpa/paq_dq_rpe	PAQ BDT Ram Read Parity Error
cpa/paq_dq_wpe	PAQ Data Queue Ram Write Parity Error

Table 5-2 Agent External Errors

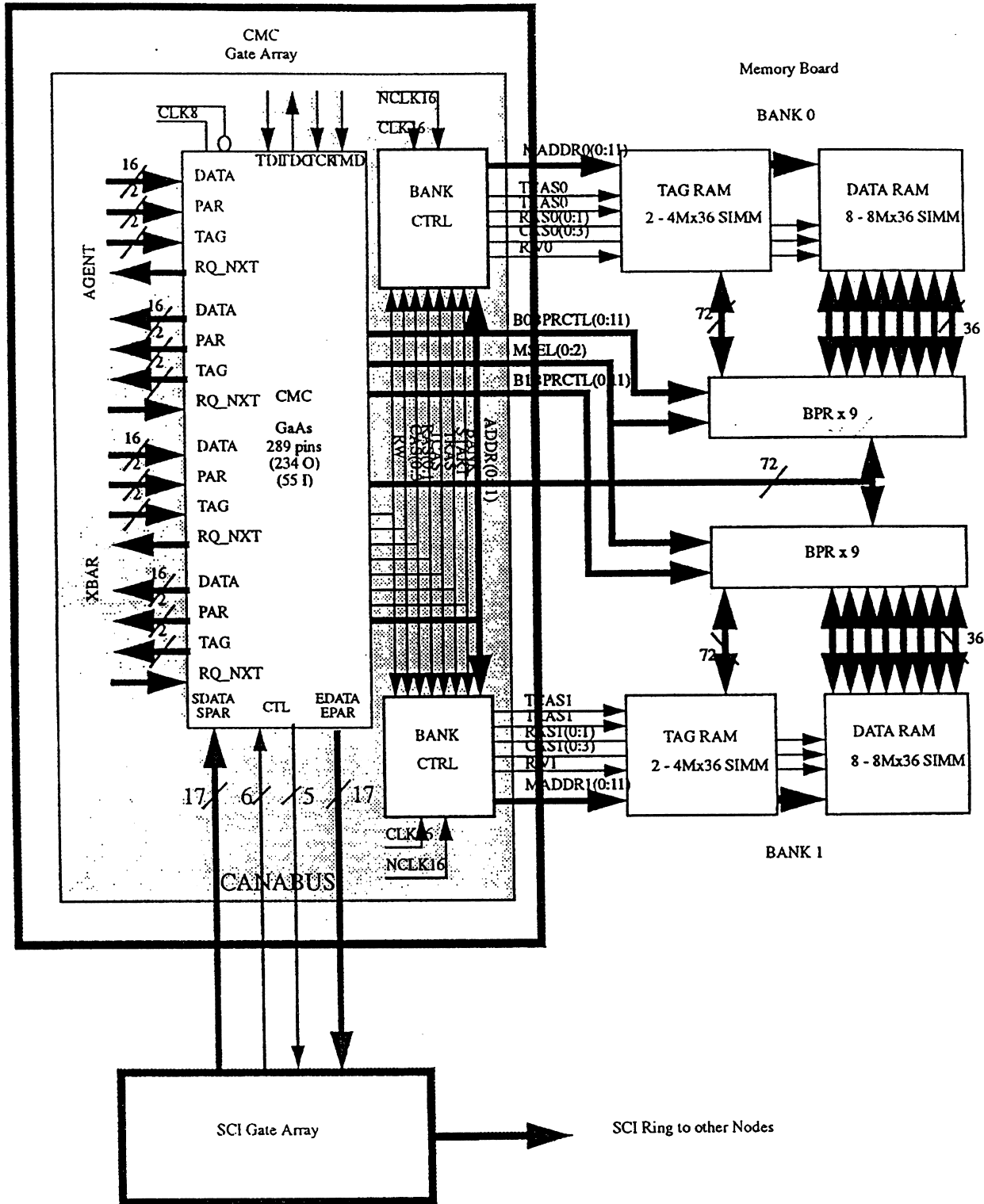
cpa/hpmch_aps_dque_rd_par_err	HIGH PRIORITY MACHINE CHECK aps_dque_rd_par_err
cpa/hpmch_aps_dque_wr_par_err	aps_dque_wr_par_err
cpa/hpmch_aps_mes_length_err	aps_mes_length_err
cpa/hpmch_csr_access_err	csr_access_err
cpa/hpmch_hu_deadlock_err	hu_deadlock_err
cpa/hpmch_hu_empty_gone_err	hu_empty_gone_err
cpa/hpmch_hu_err_cycle	hu_err_cycle
cpa/hpmch_hu_err_resp	hu_err_resp
cpa/hpmch_hu_inv_mjmn_err	hu_inv_mjmn_err
cpa/hpmch_hu_inv_resp_err	hu_inv_resp_err
cpa/hpmch_hu_invalid_err	hu_invalid_err
cpa/hpmch_software	software
cpa/lpmch_apq_mes_length_err	LOW PRIORITY MACHINE CHECK apq_mes_length_err
cpa/lpmch_apq_mjmn_req_err	apq_mjmn_req_err
cpa/lpmch_apq_non_exist_err	apq_non_exist_err
cpa/lpmch_csr_access_err	csr_access_err
cpa/lpmch_pas_dq_rpe	pas_dq_rpe
cpa/lpmch_pas_dq_wpe	pas_dq_wpe
cpa/lpmch_pas_inv_d_err	pas_inv_d_err
cpa/lpmch_pas_inv_s_err	pas_inv_s_err
cpa/lpmch_pas_rdi_s_err	pas_rdi_s_err
cpa/lpmch_roi_access_size_err	roi_access_size_err
cpa/lpmch_roi_mes_length_err	roi_mes_length_err
cpa/lpmch_software	software
cpa/paq_imiss_size_err	PAQ invalid size on lmiss
cpa/paq_mts_err	PAQ Message Transfer Sequence Error
cpa/paq_sbd_tadr_err	PAQ special BDT address Err
cpa/paq_vring_err	PAQ virtual Ring error
cpa/pas_mts_err	AS Message Transfer Sequence Error
cpa/pbi_illegal_cpa_trans	Illegal Agent initiated transaction
cpa/pbi_unimp_cpa_trans	Unimplemented P-BUS transaction
cpa/q_data_par_err	Byte parity error on CMC request port
cpa/q_tag_seq_err	Tag sequence error on CMC request port
cpa/s_data_par_err	Byte parity error on XBAR response port

5.3 CCMC Gate Array

The Convex Coherent Memory Control gate array is the interface between the memory board (MTV) and the rest of the system.

- It provides basic memory controls for the DRAM such as:
 - RAS
 - CAS
 - bank selects
 - and data interface.
- It also interfaces with
 - the CPA
 - the crossbars
 - and the SPP Coherent Interface gate array (SCI)
 - ◆ which enables node cache coherency.
- Figure 5-8 on page 84 and Figure 5-9 on page 85 are block diagrams of the CCMC.

Figure 5-8
CCMC Memory Interface Block diagram



5.3.2 CCMC Hard Errors

Table 5-3 CCMC Hard Errors

Error Name	Description
RQI_PAR_ERR	Xbar request in, Parity error
RQI_SEQ_ERR	Xbar request in, Bad head-body-tail sequence
RQI_FMT_ERR	Xbar request in, Bad packet length
RSI_PAR_ERR	Xbar response in, Parity error
RSI_SEQ_ERR	Xbar response in, Bad head-body-tail sequence
RSI_FMT_ERR	Xbar response in, Bad packet length
NCI_PAR_ERR	Node chip incoming, Parity error
NCI_FMT_ERR	Node chip incoming, Bad packet format
RQI_MISS_ERR	Retry/Unlock/Copyout missed cache
NOT_OWNER_ERR	Copyout from wrong processor
NOT_LOCKED_ERR	Retry/Unlock but lock bit not set (and should be)
RSQ_FULL_ERR	Resend Queue full
SRC_DIRTY_ERR	Line already marked dirty in requesting processor's cache
SH_LEVEL_ERR	Sharing level error, FlushP/PurgeP but line in use by SCI
SCI_TRAP_ERR	Attempt to access another node (enable for single node systems only)
RSI_BAD_RESP_ERR	Bad response received by RSI
UNREC_TAG_ERR	Unrecoverable tag error, Multi-bit ECC error while reading tags for RSI or SOCM_RQ
FLUSH_RP_ERR	Multi-bit ECC error while making Flush or Flush_RP request to SCI

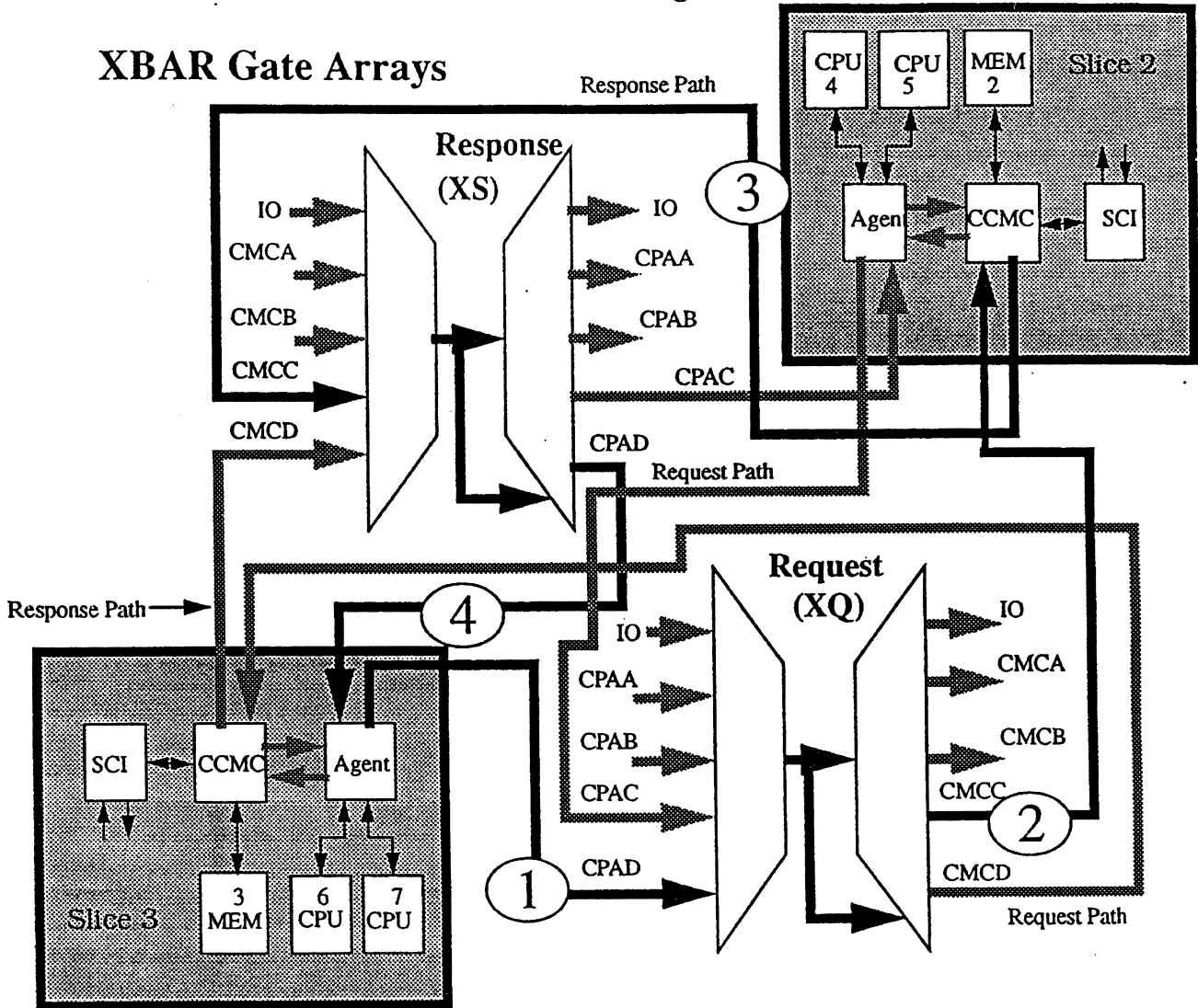
5.4 Xbar Gate Arrays

The Crossbar functions of the SPP system are implemented in two identical Xbar gatearrays.

- **XBQ** - the XBar reQuest handles all of the Requests.
- **XBS** - the XBar reSponse handles all of the responses.
- The crossbar enables memory in each slice to be shared directly with each processor within the node.
- Memory can be shared by each processor in all nodes throughout the complex, indirectly, via the SCI rings.
 - Routing from slice to slice is done by the XBAR GA on the node where the request is made.
 - ◆ IE: when accessing memory on a different slice on a different node the request only routes through the XBAR on the node where the request originated.

5.4.1 Xbar Block Diagram

Figure 5-10
Xbar Block Diagram



- (1) Agent (Port D) sends a Request to the Request Xbar (XQ).
- (2) The XQ routes the Request to the target CCMU (Port C).
- (3) The Target CCMU sends a Response to the Response Xbar (XS).
- (4) The Response Xbar routes the Response to the requesting Agent (Port D).

NOTE: Each slice has a port into the XBAR. Therefore, slice 0 = port A, etc..

5.5 MAUI Gate Array

The Mpp Agent Utility and I/O gate array is the interface between the I/O system and the Xbar.

- It acts like an Agent, but instead of interfacing with a CPU it communicates with the I/O board.
- Also, it is the link between the node's utility board and the rest of the system.
- This gate array also contains:
 - the System Control and Status Register
 - Time of Century Counter
 - Thread Execution timers
 - and Memory refresh logic.
- Figure 5-11 on page 90 is a block diagram of the MAUI.

5.5.2 MAUI Hard Errors

Table 5-4 Maui Internal Errors

maui/ebin_rd_par_err	Landmarc input channel
maui/ebin_wr_par_err	Landmarc input channel
maui/ebout_rd_par_err	Landmarc output channel
maui/ebout_wr_par_err	Landmarc output channel
maui/muri_rd_par_err	MU input channel
maui/muri_wr_par_err	MU input channel
maui/muro_rd_par_err	MU output channel
maui/muro_wr_par_err	MU output channel

Table 5-5 Maui External Errors

maui/cere_parity_err	to/from Landmarc
maui/rqi_parity_err	From the request xbar
maui/rqi_route_err	From the request xbar
maui/rqi_seq_err	From the request xbar
maui/rsi_parity_err	From the response xbar
maui/rsi_route_err	From the response xbar
maui/rsi_seq_err	From the response xbar

5.6 SCI Gate Array

The Scalable Coherent Interface (SCI) gate array is the communication link between nodes which allows quick transfers of data between the memory systems on separate nodes.

- The SCI arrays will connect the CCMC arrays to the SCI rings.
 - Data integrity is managed with CRC and is only checked at the destination.
 - There may be as many as 16 SCIs connected on a ring.
 - Each node has 4 SCI GAs, one per slice.
 - ◆ A single SCI ring will connect to the same slice on each node.
 - GSM
 - CTI

5.6.1 SCI Ring - Technical Information

- The SCI ring currently operates on a 300mhz clock.
 - Data is transferred on each clock edge.
 - **600mhz** effective clock rate.
 - ~ 667mb/s data transfer rate.
 - 18 Bit ECL Differential Unidirectional bus.

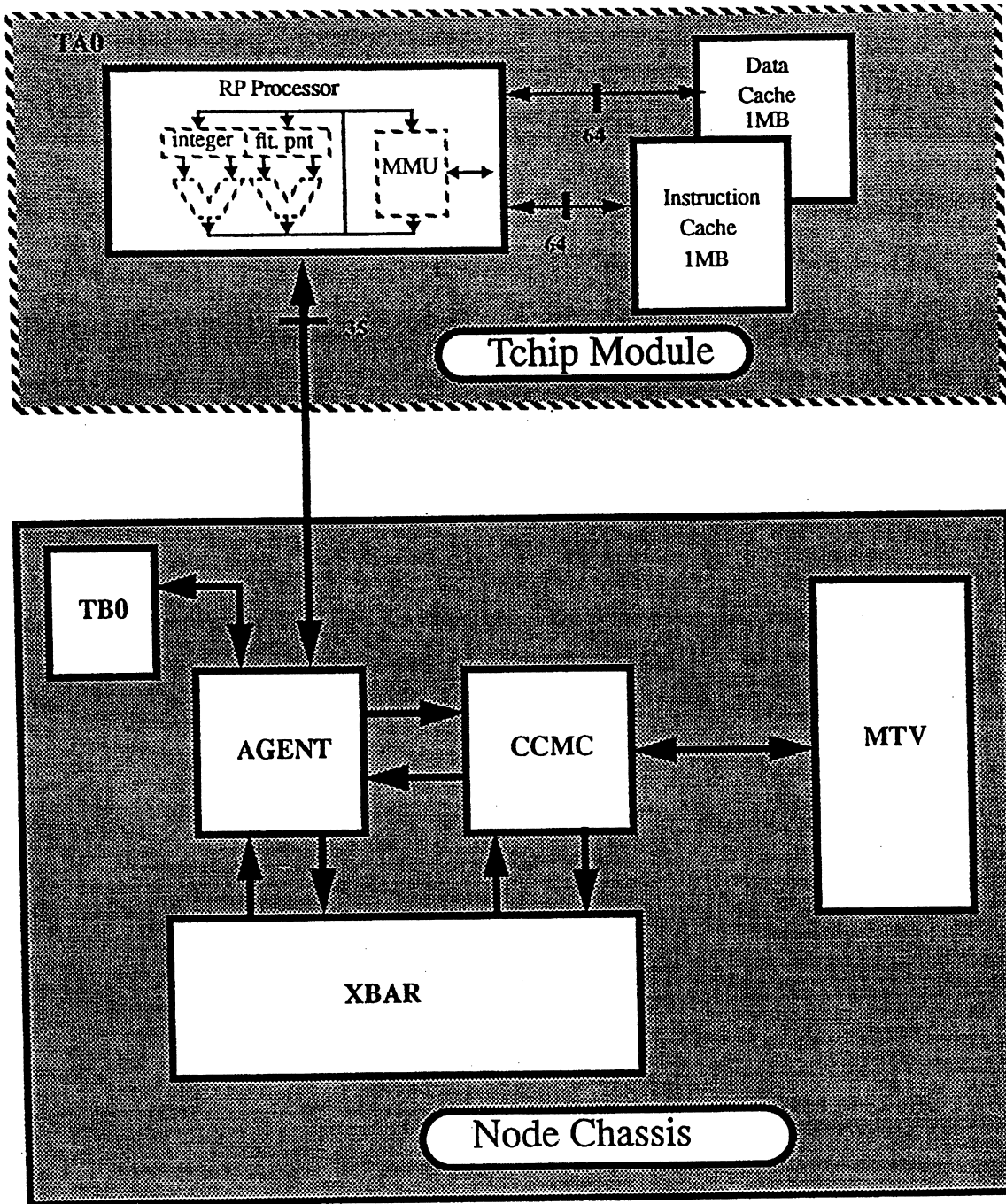
5.7 T-Chip Module: SPP-1000

- The T-Chip Module, which is manufactured by HP, is the processor board used in SPP-1000. It contains:
 - the PA-RISC Processor (7100)
 - a 1MB Data Cache
 - and a 1MB Instruction Cache.
- It communicates to the system via the Agent (CPA) gate array.
 - It communicates to the Agent through a 32 bit 100 MHZ Pbus.
- On the next page is a block diagram of the T-Chip Module.

5.8 T' (T Prime) Chip: SPP-1200/1600

- SPP-1200/1600 uses the HP PA-RISC 7200 Processor.
 - SPP-1200 contains a 256K Data and ICache.
 - SPP-1600 contains a 1Meg Data and ICache.
 - Runs at 120 MHZ.
 - Connects to the APA via the runway bus, which is a true 64 bit bus.

Figure 5-13
T-Chip Module Block Diagram

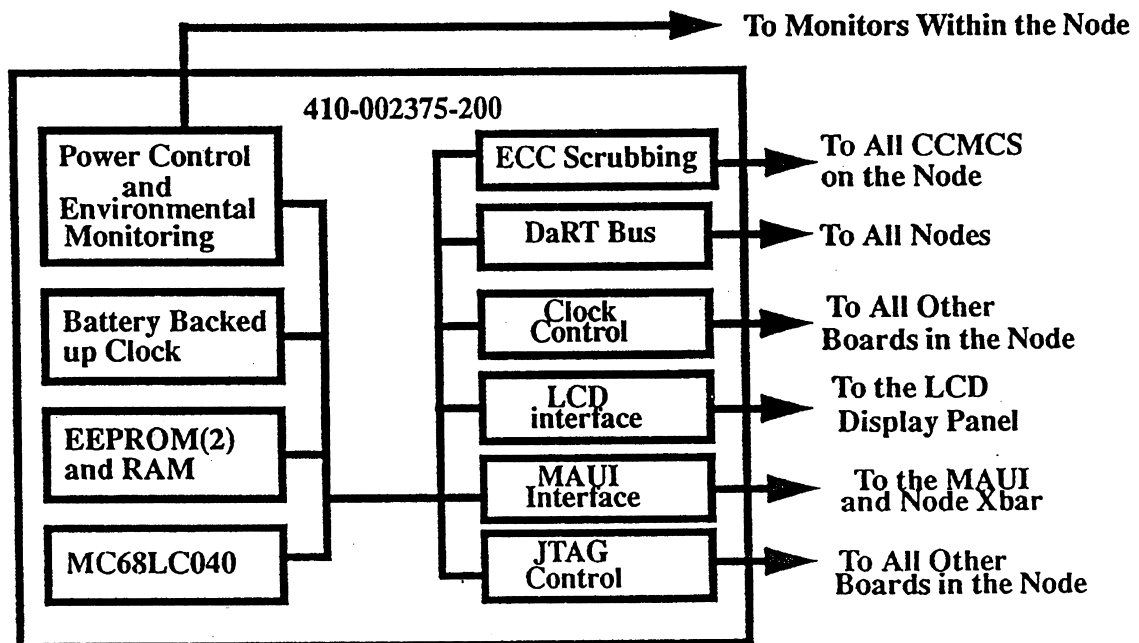


5.9 Mpp Utility board

The Mpp Utility board (MU) is the “SPU” type board for the SPP system.

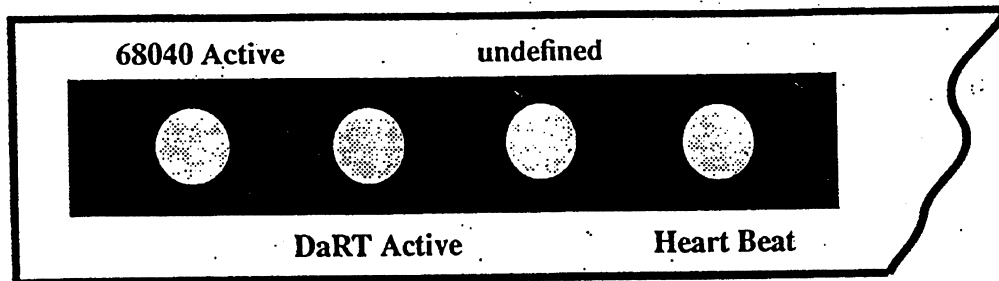
- It is the first board that the Test Station Communicates with when the system is powered on.
- The MU controls the power bricks that supply power to the rest of the node.
- It also contains the hardware configuration of the node. Figure 5-16 on page 98 is a block diagram of how the firmware, ram and nvram are arranged on the MU.

Figure 5-14
MU Block Diagram



5.9.1 MU LEDs

Figure 5-15
MU LEDs



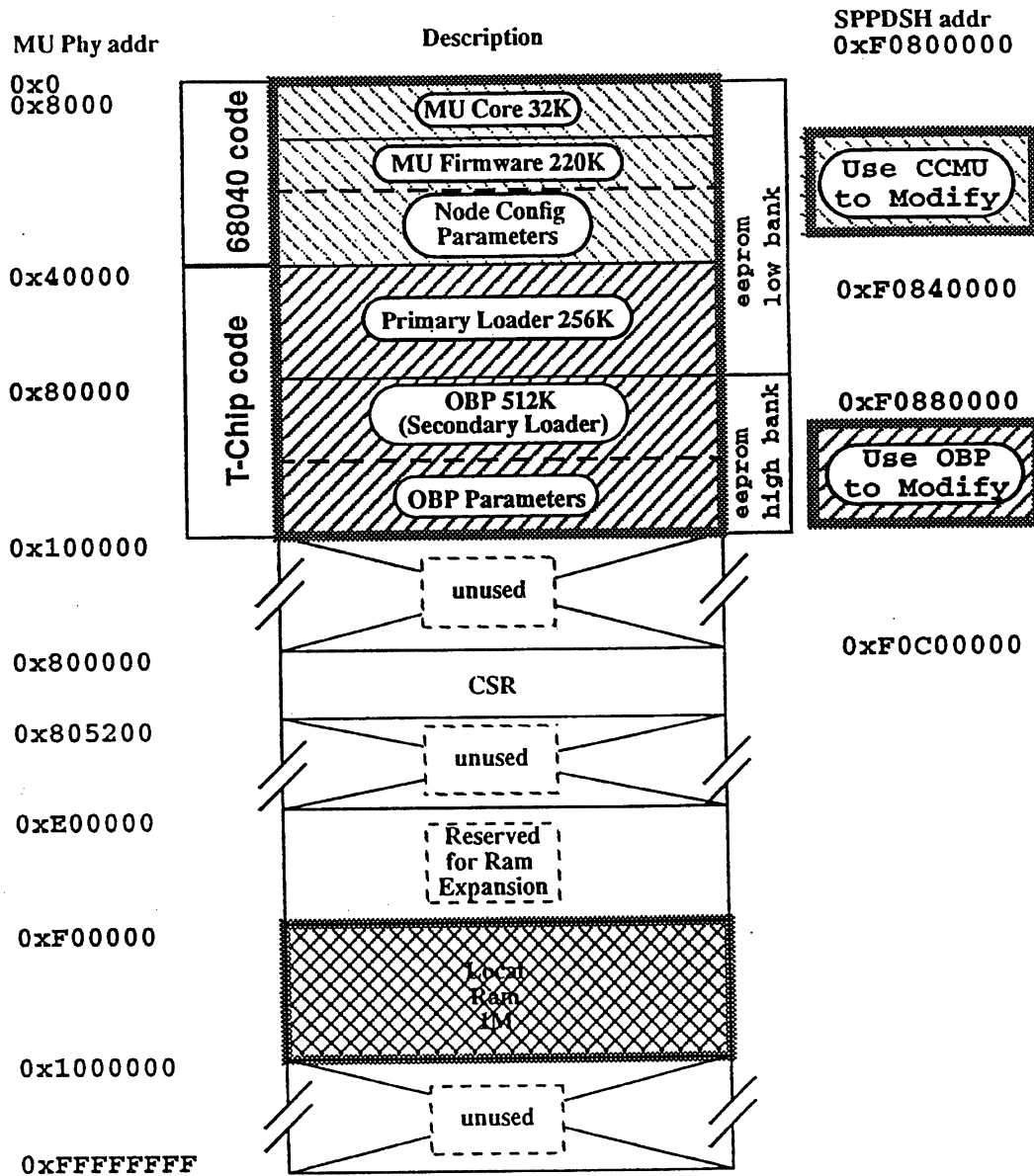
- There are fuses on the MU that should be checked on the MU if the MU LEDs do not come on.

5.9.2 MU EEPROM and DRAM

- The MU's processor runs out of DRAM.
- The DRAM gets loaded (pull) on powerup from Non-Volatile Memory.
 - The DRAM and NVM (nvram) are divided into several areas.
 - The major areas are shown in the diagram on the next page.

5.9.3 MU Memory Allocation Map

Figure 5-16
MU Memory Allocation Map



- "CCMU Camelot Configuration Management Utility" on page 151 and/or "Open Boot Program" on page 157 for more details on modifying parameters.

5.10 Memory Tag Vortex Logic Card

5.10.1 Introduction

The Memory Tag Vortex Logic Card (MTV) is the Exemplar memory card.

- There is one memory card per slice.
 - Each memory card can be accessed through the Xbar system by any CPU in the node or via the SCI bus, therefor any CPU in a complex.
 - It communicates directly with the CCMC.
 - The CCMC provides the controls for the board.
- The MTV contains Data Rams, Tag Rams and Bus Pain Relievers (BPRs).
 - 16 Data Rams per MTV for program data.
 - 4 Tag Rams per MTV, they contain important information about the data rams which is used by the CCMCs to help determine where memory is physically located.
 - The BPRs are the buffers between the CCMC and the rams.
- Memory Interleaving:
 - 32 way maximum per node (4 MTVs installed).
 - 4k page across 4 MTVs.
 - 3 MTVs/node - not recommend.
- Figure 5-17 on page 100 illustrates the physical layout.
- Figure 5-18 on page 101 is a table showing memory configuration.

Figure 5-17
MTV Block diagram

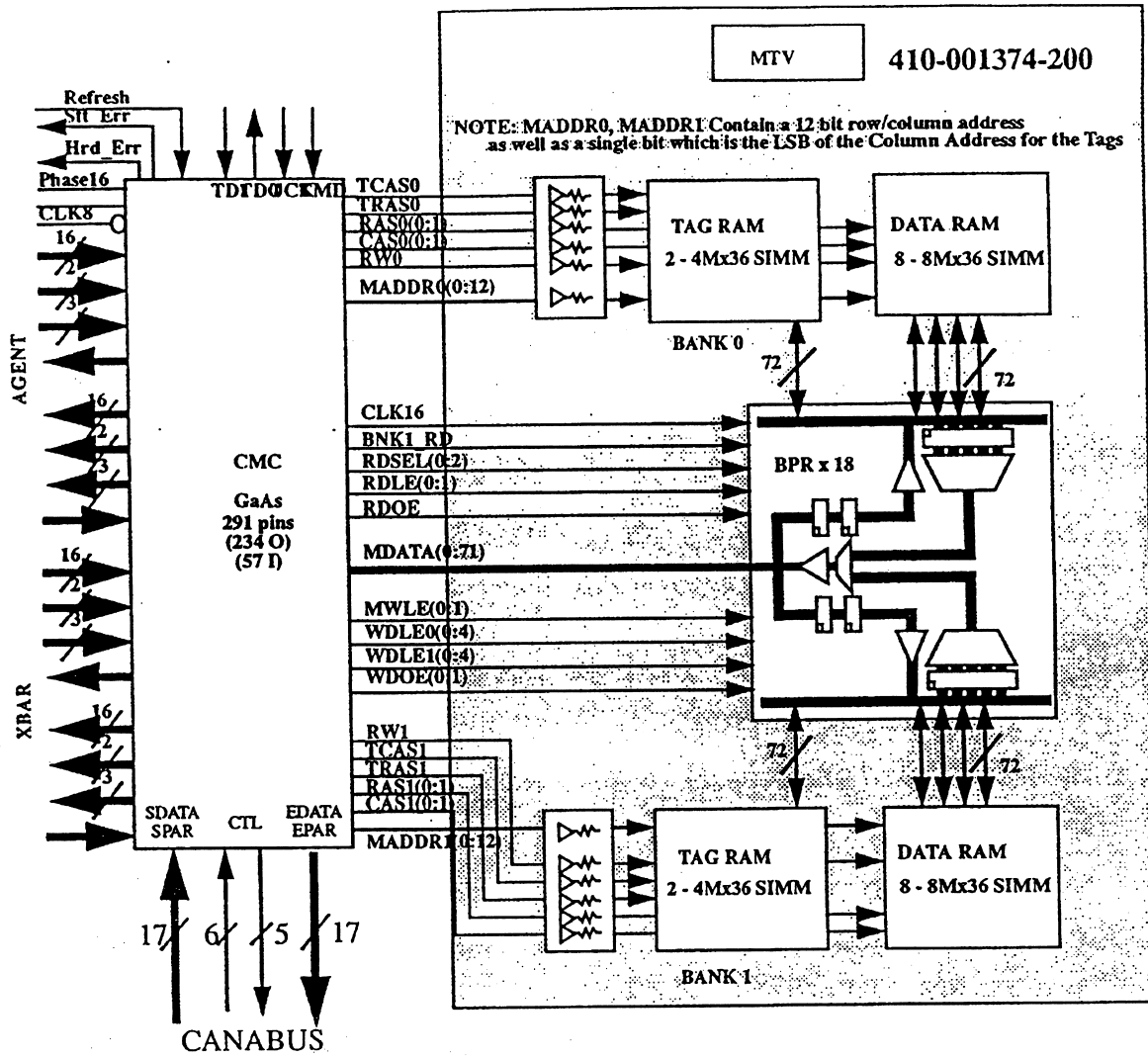
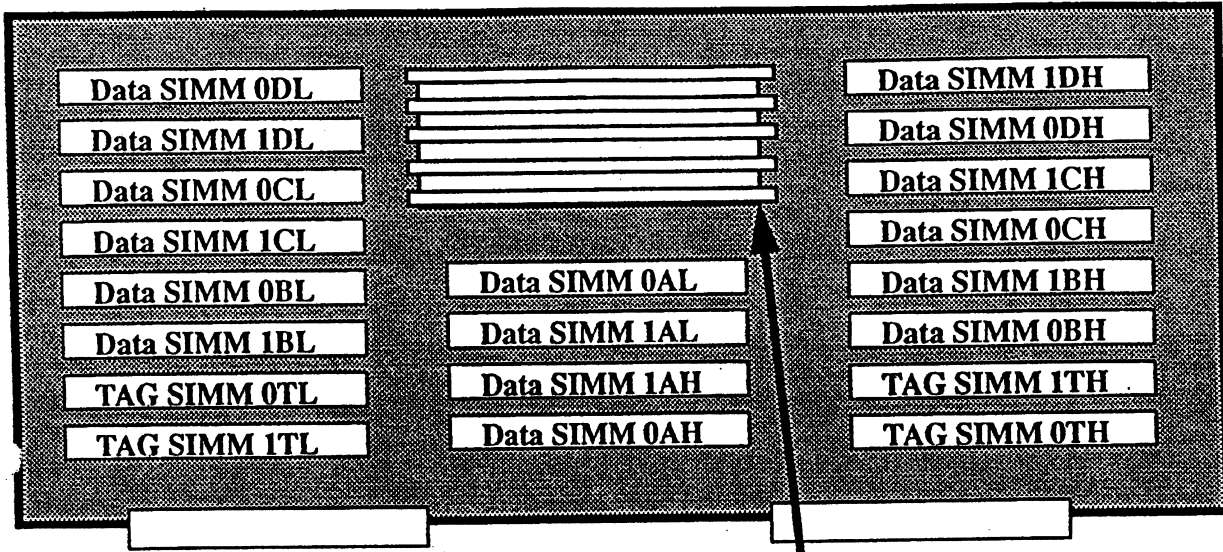


Figure 5-18
MTV Physical Layout

Top View



48VDC to 5VDC
Converter Power brick

SIMM Memory

MTV size (bytes)	SIMM size(bits)					Max Node size (bytes)	CONVEX Populated Part Number
	1Mx36	2Mx36	4Mx36	8Mx36	16Mx36		
64M	20					256M	550-000059-200
128M	4 - Tag	16 - Data				512M	550-000060-200
256M		4 - Tag	16 - Data			1024M	550-000061-200
512M			4 - Tag	16 - Data		2048M	550-000062-200
1024M				4 - Tag	16 - Data	4096M	
Max DRAM size	4Mbits	4Mbits	16Mbits	16Mbits	64Mbits		
Color Code	White	Yellow	Green	Blue	Red		
CONVEX Part Number	170-000001-001	170-000002-001	170-000003-001	170-000004-001	170-000005-001		

Figure 5-19
Memory configuration table

5.10.2 Physical memory divisions

The SPP divides its memory up into four types.

- CPU -Private Memory
 - Used by only one CPU.
 - Interleaved within a node by 64-byte cache line.
 - Since this memory is implemented on the same node as the CPU, it has very low latency - 50 clocks.
- Node-Private Memory
 - Shared by all the CPUs in a node.
 - Interleaved within a node by 64-byte cache line.
 - Node-Private memory may be implemented as an interleaving of CPU-Private memory.
 - 50 clocks.
- Near-Shared Memory
 - Shared by CPUs across nodes.
 - Interleaved within a node by 64-byte cache line.
 - Although it can be accessed by CPUs from other nodes, such accesses suffers a latency penalty.
 - CPUs accessing memory on its own node does not pay that penalty.
 - 50 - 2000 clocks.
- Far-shared Memory
 - Shared by CPUs across nodes
 - Interleaved within a node by 64-byte cache line. and across nodes by a 4-Kbyte page.
 - It allows access from all CPUs participating in an application with equal latency.
 - 200 - 2000 clocks.

Figure 5-20
Node Physical Address Space Partitioning

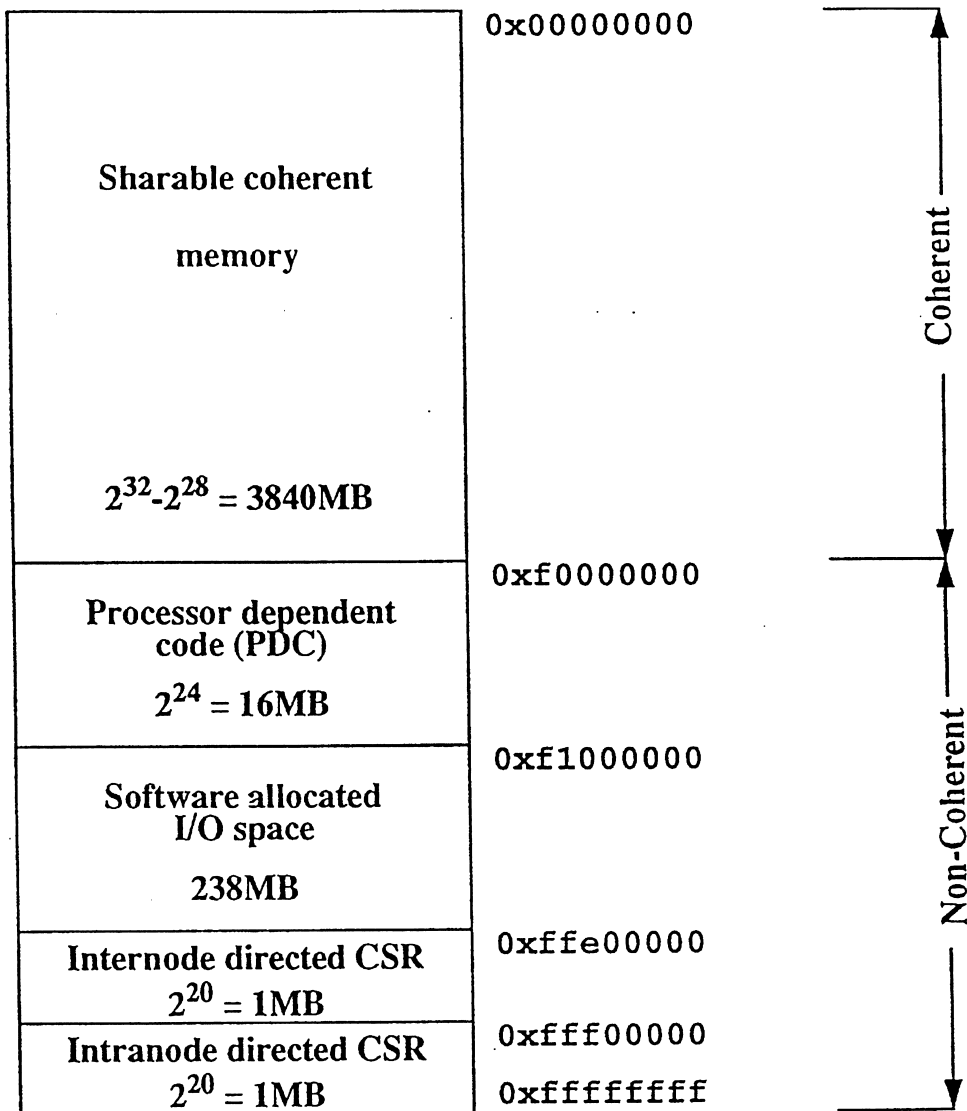
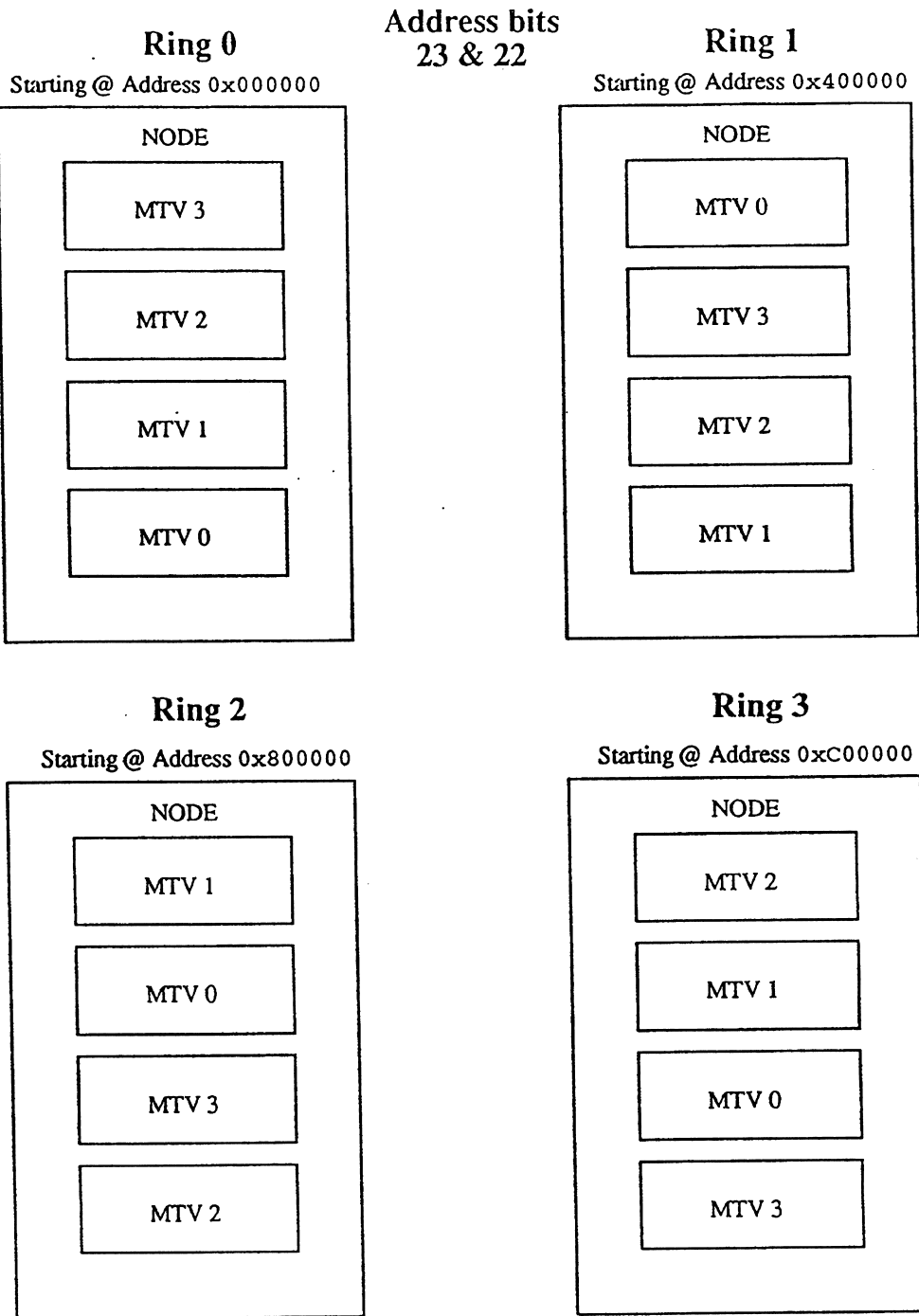


Figure 5-21
Physical Memory Addressing with Ring Bits



5.10.3 Configuring Memory with CCMU

- The SPP does not automatically size memory.
 - Whenever a memory upgrade is made or node parameters lost (new mf_fw loaded), memory board size must be manually configured using CCMU.
 - The following table defines the value of USEMEMSIZE for different size memory boards.

Table 6-1 CCMU Parameter - USEMEMSIZE

USEMEMSIZE Value	Memory Board Size
0x00000005	1024MB
0x00000004	512MB
0x00000003	256MB
0x00000002	128MB
0x00000001	64MB
0x00000000	32MB

- The steps required to change the USEMEMSIZE parameter with "ccmu" would be:
 - sppdsh> ccmu
 - ccmu> up
 - ccmu> put USEMEMSIZE 0x[]
 - ccmu> down
 - ccmu> push
 - ccmu> quit
 - sppdsh> do_reset

5.10.4 Deconfiguring Memory

Here is a table of the possible configurations that can be used when de-configuring the memory system, i.e: telling the machine the memory board is smaller than what is physically installed.

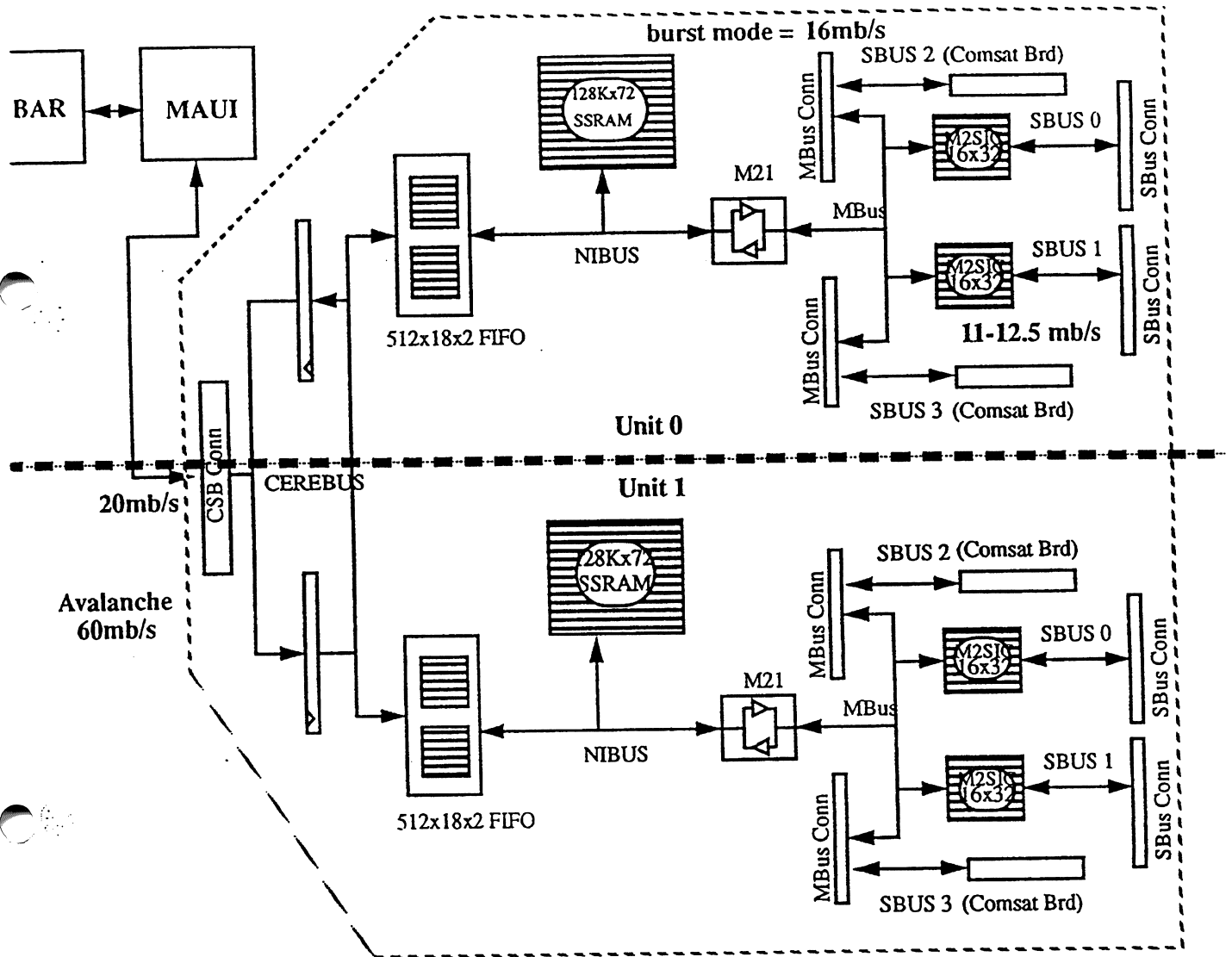
- The Table shows the value for the *USEMEMSIZE* parameter.

Original Node Mem. Size	Original Board Size	De-config. Node Mem. Size	De-config. Board Size	ccmu parm. USEMEMSIZE
256 Meg	64 Meg	N/A	N/A	N/A
512 Meg	128 Meg	256 Meg	64 Meg	1
1024 Meg	256 Meg	256 Meg	64 Meg	1
2048 Meg	512 Meg	256 Meg	64 Meg	1
2048 Meg	512 Meg	512 Meg	128 Meg	2

5.10.5 Requirements

- Memory board present on any slice using the SCI.
- At least 2 memory boards in a node.
- Any slice containing a CPU on a multi node system should also contain a MTV.

Figure 5-22
SIOP Block Diagram



6.1 I/O Chassis Types

- There are three types of I/O chassis:
 - Differential I/O Chassis
 - XA I/O Chassis
 - CD I/O Chassis
- The current production I/O chassis is the Differential I/O Chassis. This unit has replaced both the CD and XA versions of the I/O Chassis.
 - Both disk controllers and disk drives are differential.
 - A "Rancho" board is used to convert from differential to single ended for the dat tape drive.
- Both the XA and CD chassis used single ended disk drives.
 - With the XA I/O Chassis, differential disk controllers are used in order to extend cable lengths.
 - ◆ Ancots boards are used to convert from differential to single ended cables once the bus reaches the I/O chassis.
 - The CD version of the I/O chassis are only found inside CD nodes.
 - ◆ Single ended disk controllers are used with the CD I/O Chassis.

6.2 Boards Chassis Support Board

- The boards in the peripheral chassis are:
 - **CIOBP** - Convex IO backplane
 - **CIOIR** - Convex IO Right address card
 - **CIOIL** - Convex IO Left address card
 - **CIOPC** - Convex IO Power Controller
 - **Ancot** - SCSI Differential to single ended converter
 - **CIODA** - Interface card for disk drive to XA IO chassis
 - **Rancho Board** - SCSI Differential to single ended converter

6.3 Peripheral Configurations

- Xa High Performance
- Xa High Capacity
- CD configuration
- I/O Expansion chassis XA
- I/O Expansion chassis CD
- Differential Chassis

6.4 XA

6.4.1 High Performance

Figure 6-1
The High Performance Peripheral chassis

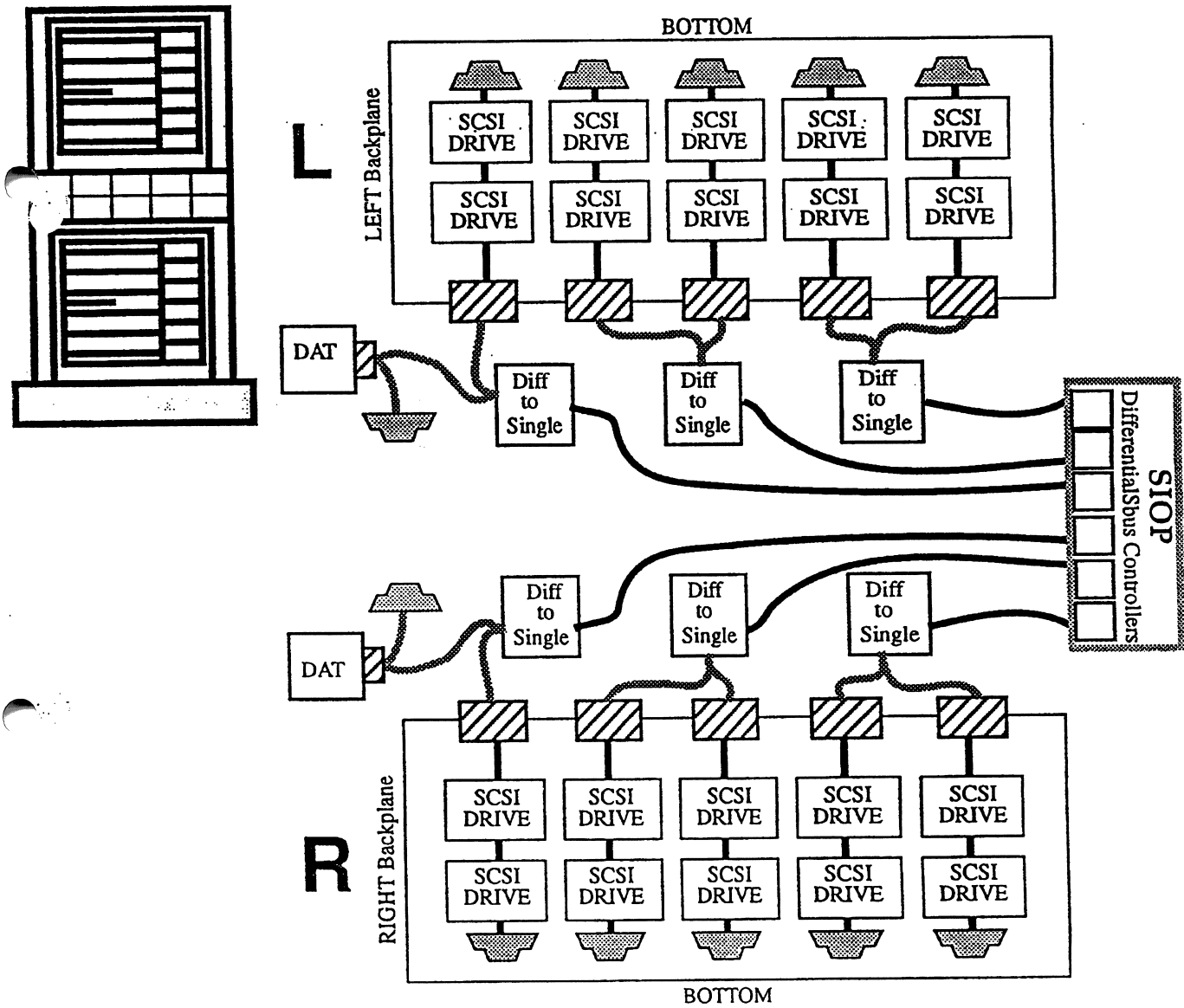
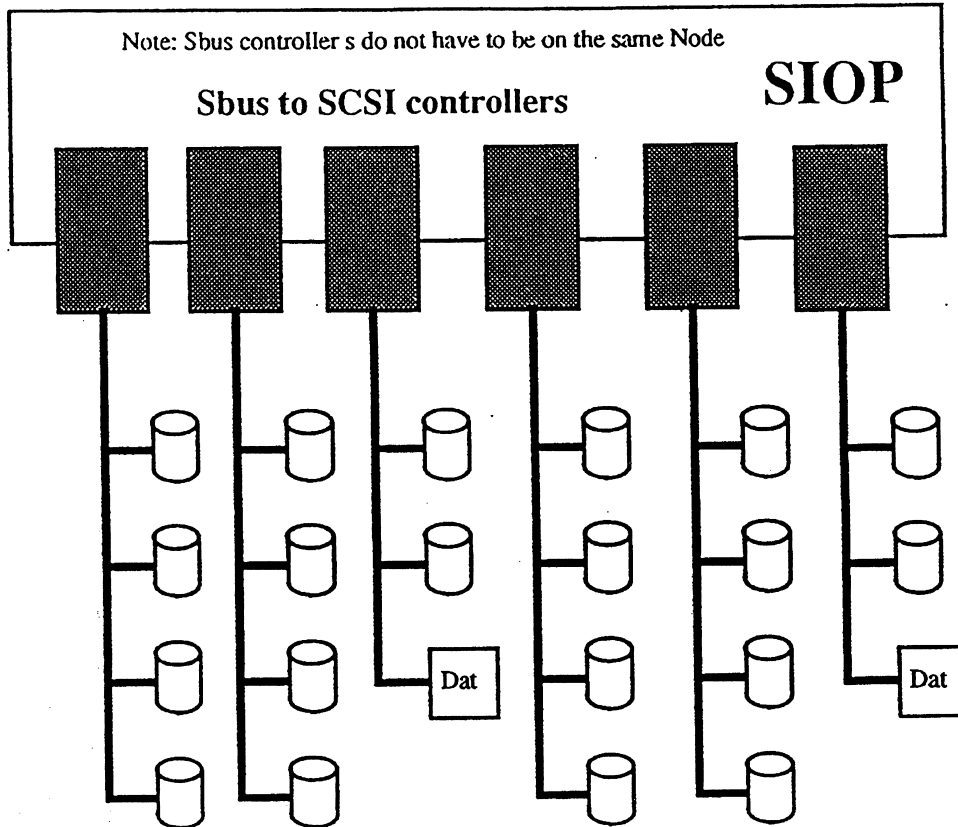


Figure 6-2
The High Performance Block Diagram



6.4.2 High Capacity

Figure 6-3
The High Capacity Configuration

BOTTOM

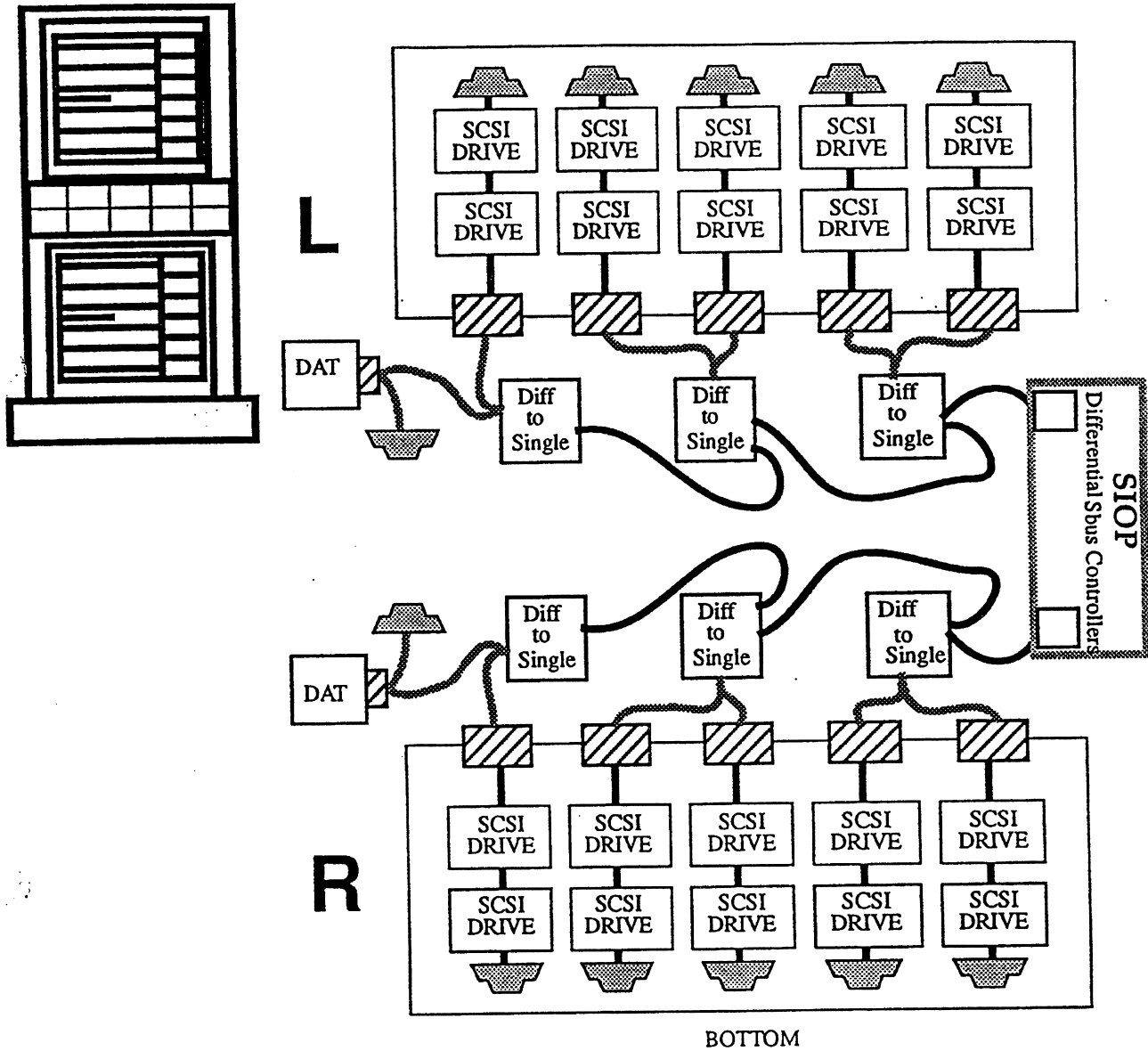
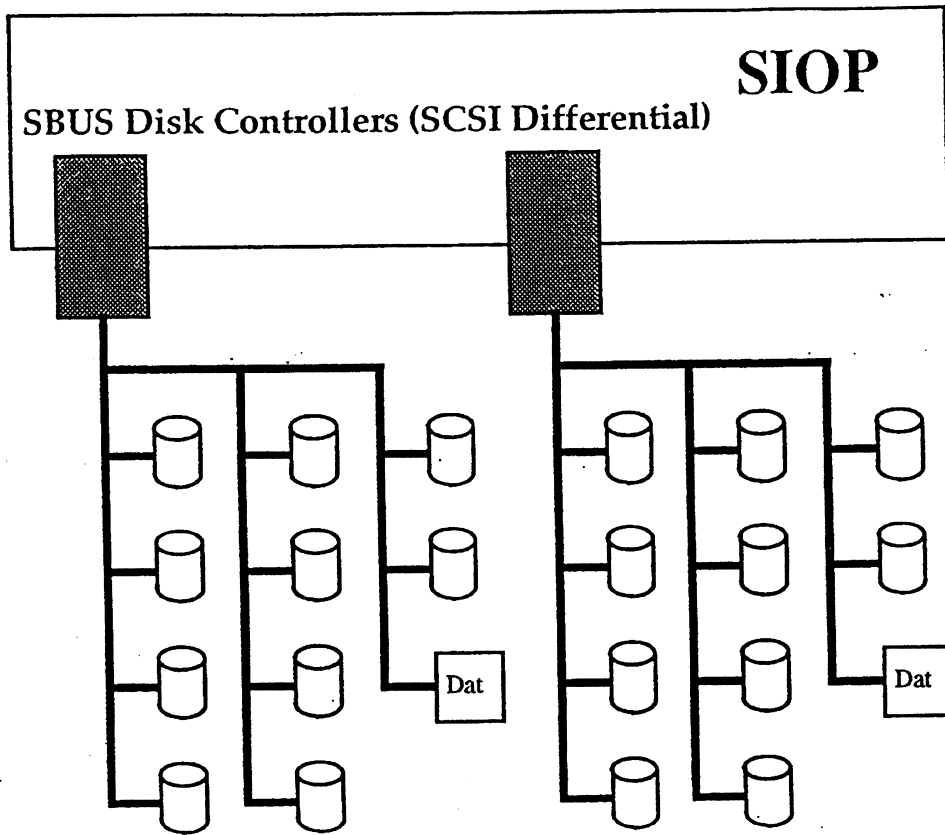
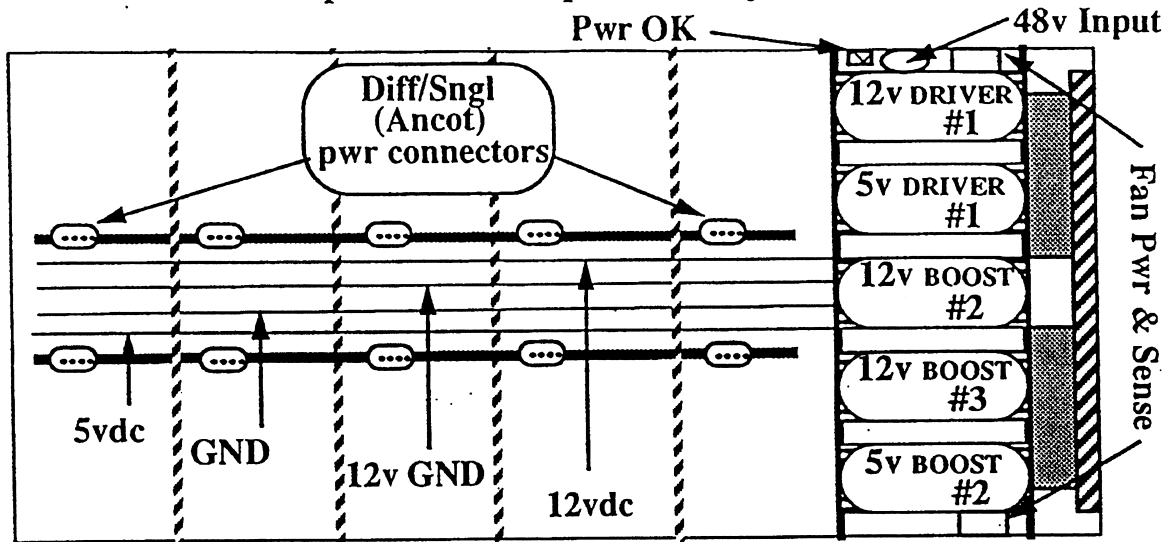


Figure 6-4
The High Capacity Block diagram



6.4.3 Power system

Figure 6-5
Peripheral cabinet power subsystem



CIOPC 410-001442-200 IO Chassis Power Controller

410-001443-200 BD ASSY, CIOPB, 5V, DRIVER

410-002443-200 BD ASSY, CIOPB, 12V, DRIVER

410-003443-200 BD ASSY, CIOPB, 5V, BOOSTER

410-004443-200 BD ASSY, CIOPB, 12V, BOOSTER

6.4.4 Slot Addresses

Figure 6-6
SCSI ID's (XA Chassis)

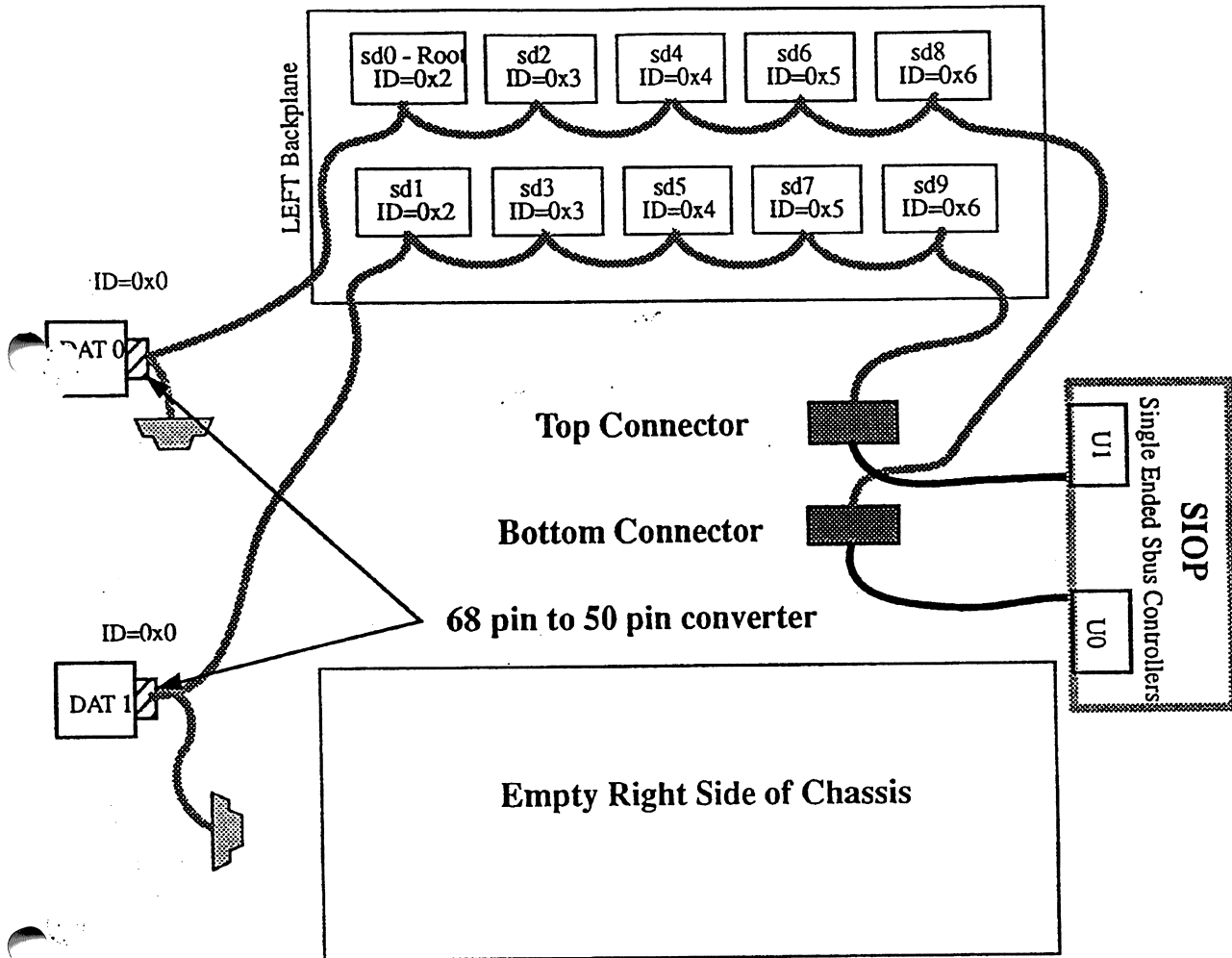
SCSI IDs

LEFT Backplane TOP of backplane					dat 1 ID=0x0
sd10 ID=0x2	sd12 ID=0x6	sd14 ID=0xb	sd16 ID=0x9	sd18 ID=0x4	FRONT of cabinet
sd11 ID=0x3	sd13 ID=0x8	sd15 ID=0xc	sd17 ID=0xa	sd19 ID=0x5	

RIGHT Backplane TOP of backplane					
dat 0 ID=0x0	sd0 (root) ID=0x2	sd2 ID=0x6	sd4 ID=0x4	sd6 ID=0x9	sd8 ID=0xb
FRONT of cabinet	sd1 ID=0x3	sd3 ID=0x8	sd5 ID=0x5	sd7 ID=0xa	sd9 ID=0xc

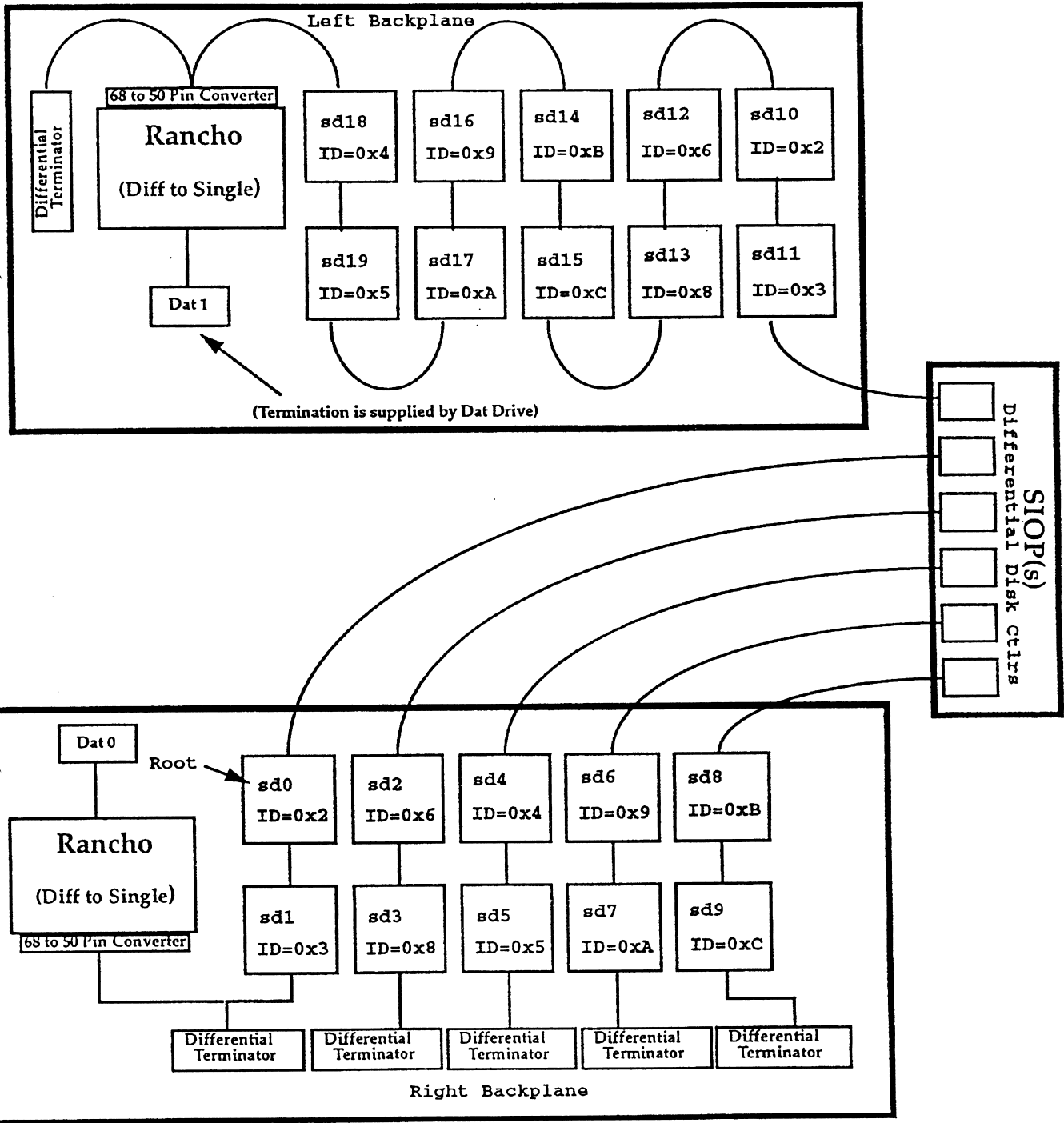
6.5 CD Configuration

Figure 6-7
CD cable diagram



6.6 Differential Chassis

Figure 6-8
Differential I/O Chassis



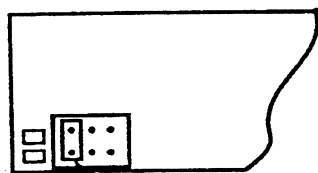
6.7 Disk Drives - Strapping

- Disk drive ID strapping is only necessary for drives installed in the CD I/O chassis using a single ended scsi controller.

6.7.1 DEC (2 GIG)

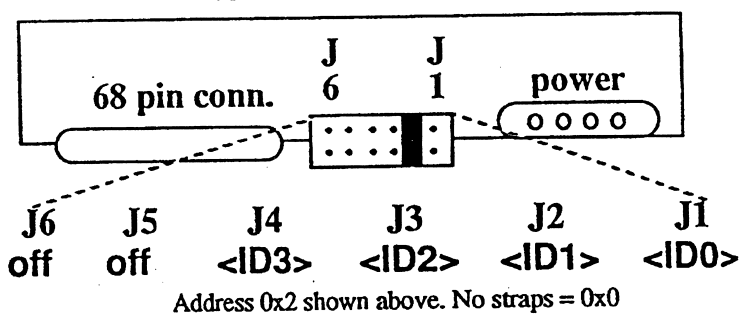
Figure 6-9
Disk Drive Strapping (DEC DSP3210W 2)

Front View of the Drive



Pwr on spin-up

View from the back



6.7.2 Seagate (4 GIG)

- Option connector:

```

+-----+
|                                             J01 ::|
|                                             |
|               option      secondary ::|
|               jumpers     jumpers   ::|
|               J    J           power  ::|
|-----| 6    1    /-----\ |
+--\__68 pin conn.__/---:---:---:---:---|_0_0_0_0|---+

```

```

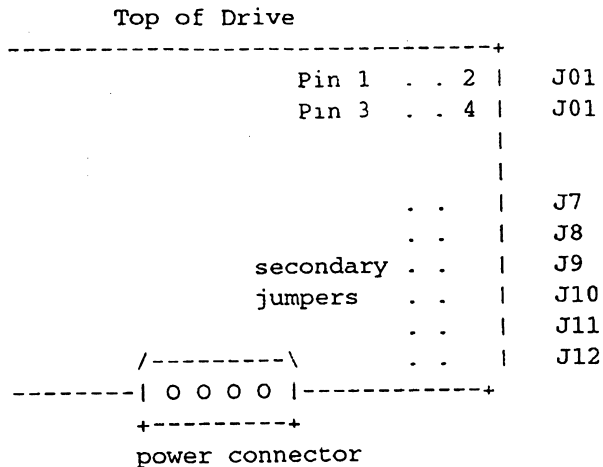
      Jumper Settings on Option Connector
      J6      J5      J4      J3      J2      J1
      off     off     <ID3>  <ID2>  <ID1>  <ID0>

```

- J6 controls spindle sync and should be left off under normal conditions.
- J5 controls remote LED connection and should also be left off.
- For use in an SPP/XA Chassis, leave jumpers J1-J6 OUT.
- J01 - termination and termpower:
 - In single-ended drives, we want the drive to receive and provide terminator power to the SCSI bus

Jumper settings for J01:

Pin 1 to Pin 3	Drive provides termpower to the SCSI bus. *****This jumper should be set.
Pin 1 to Pin 2	Drive provides its own termpower
Pin 2 to Pin 4	Termpower for the drive from SCSI bus. *****This jumper should be set.
Pin 1 to Pin 3 and Pin 2 to Pin 4	Terminator power to the SCSI bus and drive. (Both should be set for default operation)



- Secondary jumper settings:
 - Should all be set to off.

- J12** **Enable Termination** - Jumper off to disable drive termination, Jumper on to enable drive termination. Note: Most of the applications have an external Fast/Wide terminator, in which case this jumper should not be installed.
- J11** **Reserved**
- J10** **Parity Check** - Leave jumper off to enable parity check. Jumper on to disable parity checking.
- J9** **Motor Start** - Jumper on to make drive wait for Start Unit command before starting spindle motor. Jumper off causes drive to start based on J8 setting.
- J8** **Delay Motor Start** - Jumper on to make drive start delay equal to the SCSI ID multiplied by 10 seconds. For example, for SCSI ID=2 the drive will delay for 20 seconds before starting.
- J7** **Write Protect** - Jumper on to disable writing. Leave Jumper off to enable writing.

6.7.3 Dat Drive - XA/CD I/O Chassis

- Internal Terminators
 - Remove the two SIP terminators on the left side, which will disable the termination on the drive.
 - Also, make sure the terminator power jumper is installed.
- Switch Settings

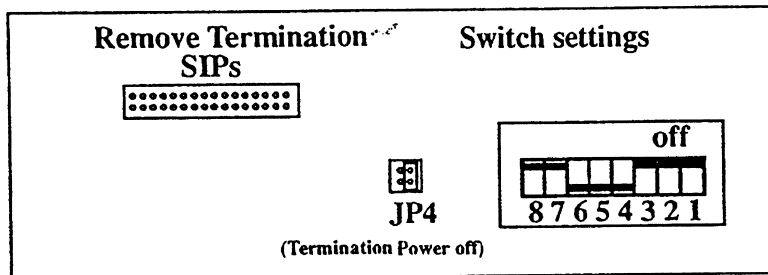
Switches S1, S2, and S3 control the SCSI device address.

S3	S2	S1	SCSI Device Address
off	off	off	0
off	off	on	1

The following is a list for the rest of the jumper settings:

Switch Setting Meaning	S4	S5	S6	S7	S8
	ON	ON	OFF	OFF	OFF
	MRS	Parity	Compression	reserve	Self-test
	On	On	Enabled		

Figure 6-10
DAT Drive strapping

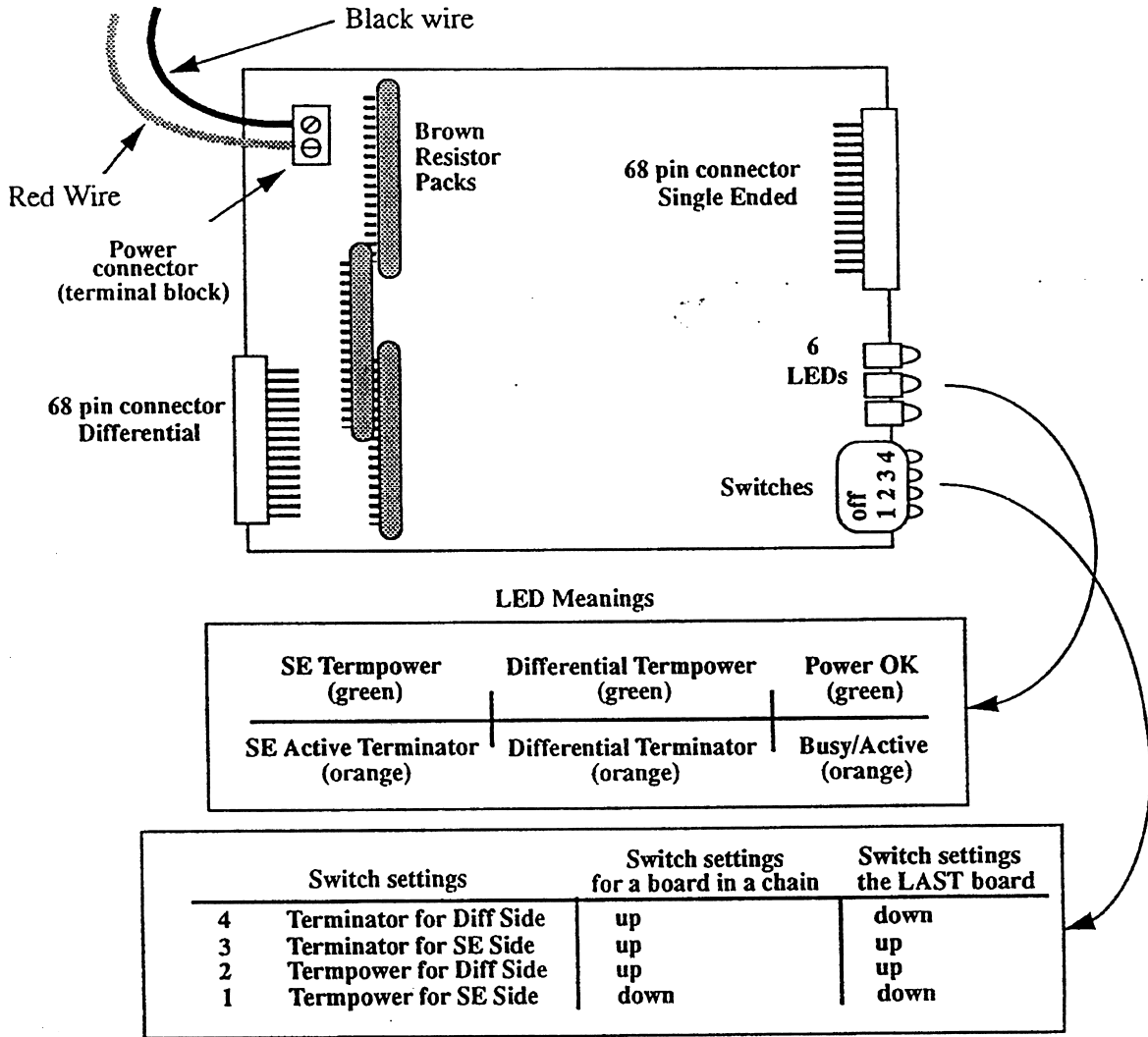


6.7.4 Dat Drive - Differential I/O Chassis

- A new dat drive is being supplied with the differential chassis.
- This dat drive will supply it's own termination.
 - Pins 11-12 must be jumpered to supply termination.

6.7.5 Ancot Card

Figure 6-11
Differential to Single Ended Converter Card



6.8 IO Address Reference Sheet (nvalias format)

Addresses as seen in the OBP window:

- *SIOP in node 0*
/mbus@0
- *SIOP in node 1*
/mbus@10000
- *SIOP in node 15*
/mbus@f0000
- *SIOP node 1 unit 0*
/mbus@10000, ffec0000
- *SIOP node 1 unit 1*
/mbus@10000, ffed0000
- *SIOP node 0 unit 0 sbus slot 0*
/mbus@0, ffec0000/sbus@f, fcffff00
- *SIOP node 0 unit 0 sbus slot 1*
/mbus@0, ffec0000/sbus@f, fdffff00
- *SIOP node 0 unit 1 sbus slot 0*
/mbus@0, ffed0000/sbus@f, fcffff00 (slot 2 - feffff00)
- *SIOP node 0 unit 1 sbus slot 1*
/mbus@0, ffed0000/sbus@f, fdffff00 (slot 3 - fffffff0)
- *SIOP node 0 unit 0 sbus slot 0 Fast/Wide SCSI*
/mbus@0, ffec0000/sbus@f, fcffff00/Convex, afws@1, 10000
- *SIOP node 0 unit 1 sbus slot 0 FDDI CRESENDO*
/mbus@0, ffed0000/sbus@f, fcffff00/CRES, cddi@1, 400000
- *SIOP node 0 unit 1 sbus slot 0 Fast/Wide SCSI SCSI ID 0x2 partition a*
/mbus@0, ffed0000/sbus@f, fcffff00/Convex, afws@1, 10000:narrow/sd@2, 0:a

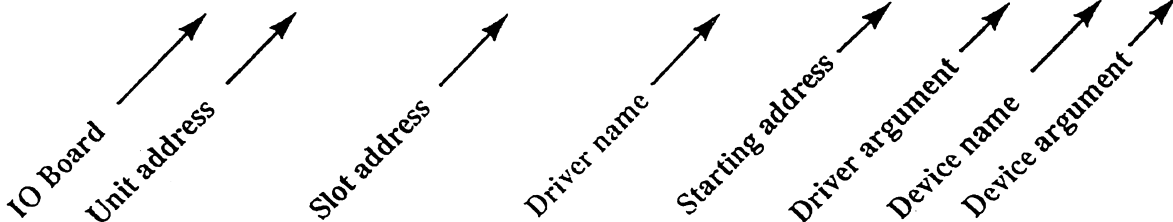
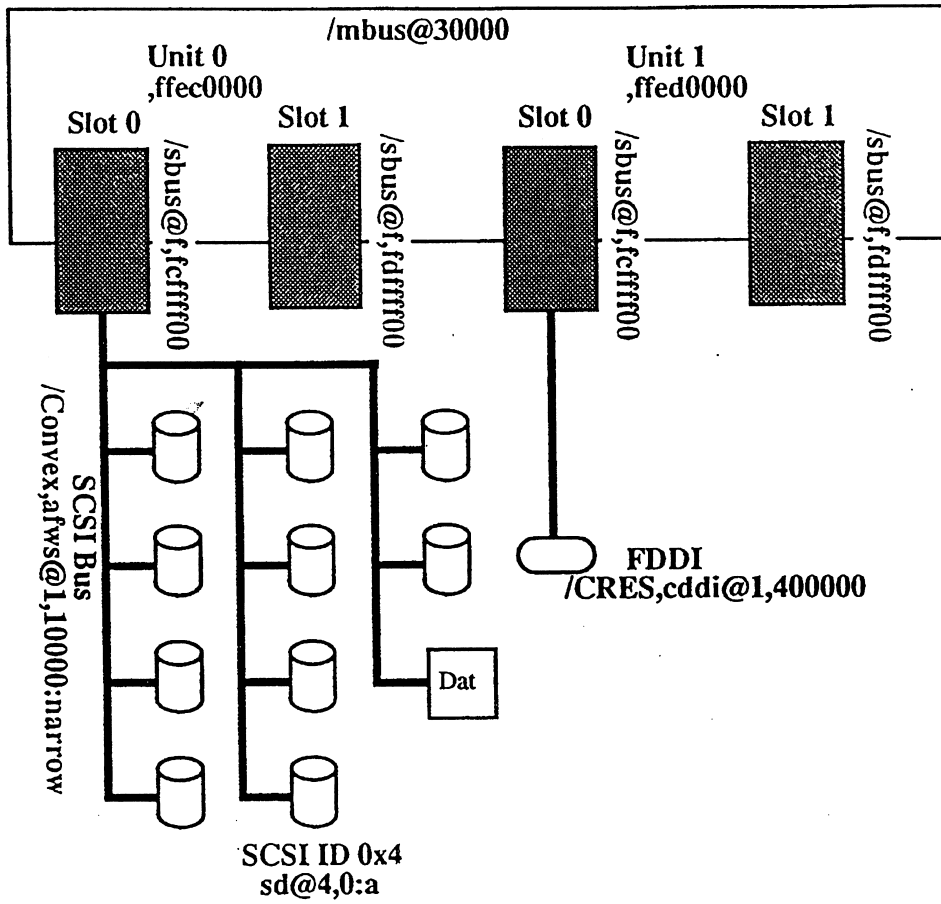


Figure 6-12
IO Address Figure



6.9 OBP Setup Params for IO Devices (mkmap)

6.9.1 Dat Drive Examples

The following (in bold) would be entered in the OBP window to set up 4 dat drives on a complex.

- 1 Dat on node 0
 - 0 node
 - **mkmap 0 /mbus@0,ffec0000/sbus@f,fcffff00/Convex,afws@1,10000/st@0,0**
- 1 Dat on node 1
 - 1 node
 - **mkmap 1 /mbus@10000,ffed0000/sbus@f,fdffff00/Convex,afws@1,10000/st@0,0**
- 2 Dats on node 2
 - 2 node
 - **mkmap 2 /mbus@20000,ffec0000/sbus@f,fcffff00/Convex,afws@1,10000/st@0,0**
 - **mkmap 3 /mbus@20000,ffec0000/sbus@f,fdffff00/Convex,afws@1,10000/st@0,0**

6.9.2 FDDI Example

- FDDI on node 0, unit 1, sbus 0
 - **mkmap -n 0 /mbus@0,ffed0000/sbus@f,fcffff00/CRES,cddi@1,400000**

NOTE: For mkmap entries to take effect, the OBP must be reset.

Internal Communications

7

The communication between functional blocks in the Exemplar is done by a complex system of transactions. These transactions are accomplished with a **packet system**. In this section, packet descriptions and formats will be detailed.

7.1 The Physical data path

The physical data path between functional blocks consist of the following:

- Data bits 0-15
 - Unidirectional bus, it is the communication link between functional blocks, excluding the MTV.
 - The bus contains:
 - ◆ Routing information
 - ◆ control information
 - ◆ address information
 - ◆ and of course... data.
 - There are other informational packets that travel on this bus that will be discussed late.

- Data Parity bits 0-1
 - These uni-directional lines guarantee that the data bus contains "valid" data.
 - If the data lines drop or pick a bit, the parity checkers at the input of each functional block detect the error.
 - The report (in the event_log) will designate it a parity error.

- Tag bits 0-2
 - The Tag bits are used to identify the sequence of the transactions. The sequence possibilities are:
 - ◆ **Head** - The Beginning of the transaction containing the routing information.
 - ◆ **Body**- The main body of information, varies in size.
 - ◆ **Tail** - The end of the transaction.
 - ◆ **Error** - Containing error information.

- Tag Parity (1 bit)
 - Tag parity is used to monitor the tag bits.

7.2 Packet Types

There are two kinds of packets: request packets and response packets.

7.2.1 Request Packets

The request packets are divided into two classes:

- Major Packet Types
 - BASIC
 - RTRY
 - UNLOCK
 - UNLOCK_WR
 - COPY_OUT
 - RD_IND
 - INV
 - TBL_INV
 - NRD_SB
 - NRD_64
 - N_HDR
 - SCRUB
- Minor Packets
 - (omitted at this time)

7.2.2 Response (Send) Packets

The send packets are also divided into two classes:

- Major Packet Types
 - BASIC_RES
 - PRIMARY_RES
 - GENERAL_RES
 - RESEND
 - NAK
 - NAK_FRZ
 - RTRY_MISS
 - IND_MISS
 - GONE
 - MLR
 - PLR
 - NLOCK_RES
 - NRD_RES_SB
 - NRD_RES_64
 - NWR_RES
 - ERROR
 - ENQUEUED
- Minor Packet Types
 - DRD_SH
 - IRD_SH
 - DRD_PR
 - LDC
 - DFLUSHP
 - DFLUSHK
 - DFLUSHN
 - DFLUSHG

- DPURGEP
- DPURGEK
- DPURGEN
- FINC
- FDEC
- FCLR
- SEND
- PRE_RD
- NPRE_RD
- PRE_WR
- IFLUSH
- SHARED_INV
- PRIVATE_INV
- ITLB
- DTLB
- SB
- PRIVATE_INV_HIT
- PRIVATE_INV_MISS
- SCI_INV
- COPY_OUT_RES
- UNLOCK_RES

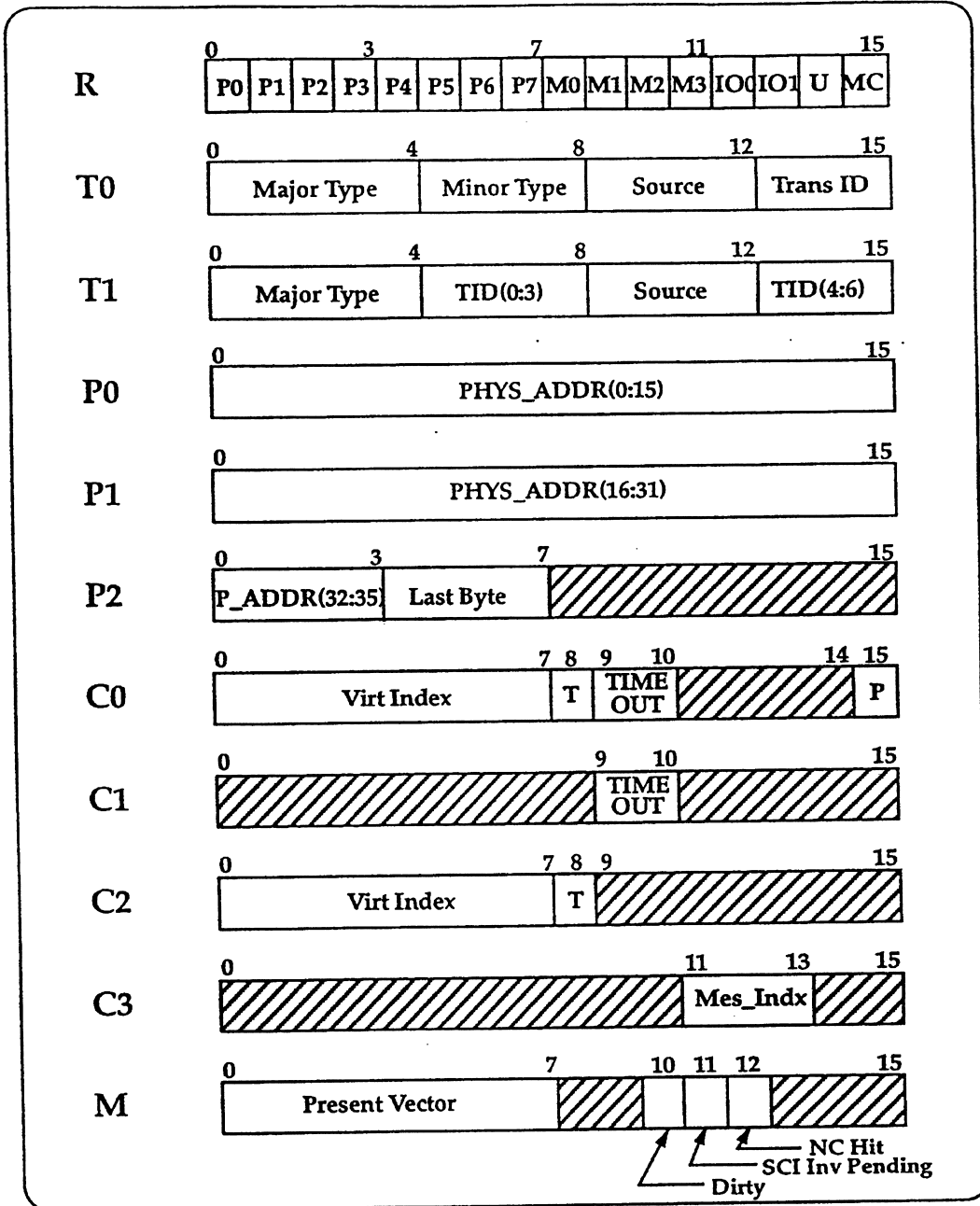
7.3 Packet Structures

In each packet information is formatted depending on the kind of transaction being performed.

- A Routing word marks the beginning of every packet.
- The transaction type follows the routing word.
- Depending on the type of transaction, packet words are formatted.
- Below is a key to the following diagrams:
 - R = Routing word
 - T = Type word
 - P = Physical Address
 - C = Control Word
 - M = Memory Coherency response word
 - V = Virtual Index
 - D = Data word
 - E = Error word

The following figure diagrams the format word structure.

Figure 7-1
Packet Word Format



Packet word formats (Continued)

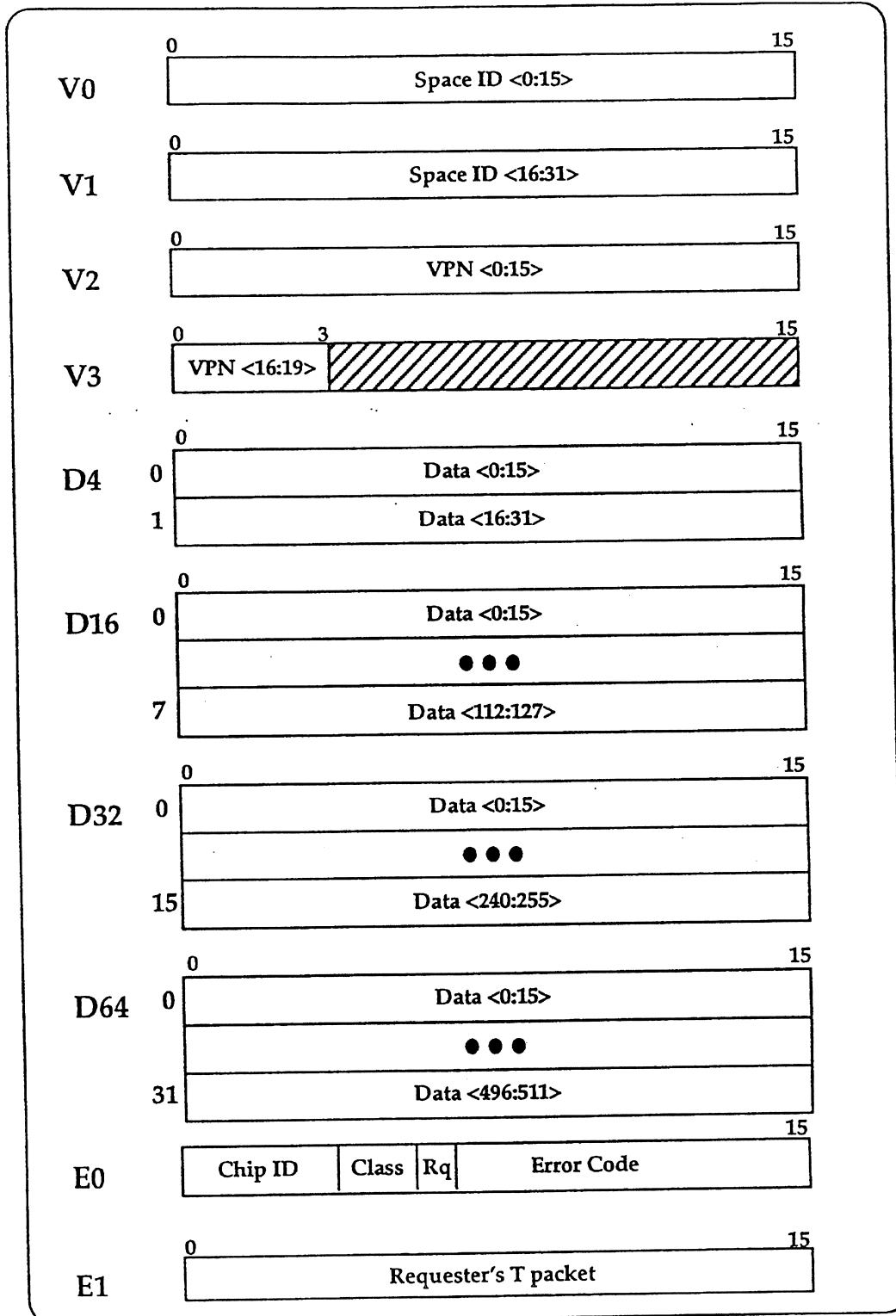


Table 7-1 Typical packet formats (request packets)

Major Type	Minor Type	Packet Format
BASIC	Point to point requests	
	DRD_SH	R, T0, P1, P0, C0
	IRD_SH	R, T0, P1, P0, C0
	DRD_PR	R, T0, P1, P0, C0
	LDC	R, T0, P1, P0, C0
	DALLOC	R, T0, P1, P0, C0
	DFLUSHP	R, T0, P1, P0, C1
	DFLUSHK	R, T0, P1, P0, C1
	DFLUSHN	R, T0, P1, P0, C1
	DFLUSHG	R, T0, P1, P0, C1
	DPURGEP	R, T0, P1, P0, C1
	DPURGEK	R, T0, P1, P0, C1
	DPURGEN	R, T0, P1, P0, C1
PRIMARY	Point to point requests	
	FINC	R, T0, P1, P0, C1
	FDEC	R, T0, P1, P0, C1
	FCLR	R, T0, P1, P0, C1
	SEND	R, T0, P1, P0, C3
	PRE_RD	R, T0, P1, P0, C0
	NPRE_RD	R, T0, P1, P0, C0
	PRE_WR	R, T0, P1, P0, C0
RTRY	Point to point requests	
	DRD_SH	R, T0, P1, P0
	DRD_PR	R, T0, P1, P0
	LDC	R, T0, P1, P0
	DALLOC	R, T0, P1, P0
	DFLUSHK	R, T0, P1, P0
	DPURGEK	R, T0, P1, P0

Table 7-1 Typical packet formats (request packets)

Major Type	Minor Type	Packet Format
UNLOCK	Point to point requests to memory	
	DRD_PR	R, T0, P1, P0
	DALLOC	R, T0, P1, P0
	DFLUSHK	R, T0, P1, P0
	DPURGEK	R, T0, P1, P0
UNLOCK_WR	Point to point requests to memory	
	DRD_SH	R, T0, P1, P0, D32
	LDC	R, T0, P1, P0, D32
	DFLUSHK	R, T0, P1, P0, D32
COPY_OUT	Point to point request to memory - R, T0, P1, P0, D32	
RD_IND	Point to point requests to a processor	
	DRD_SH	R, T0, P1, P0, C2
	DRD_PR	R, T0, P1, P0, C2
	LDC	R, T0, P1, P0, C2
	DFLUSHK	R, T0, P1, P0, C2
INV	Multicast requests to selected processors	
	IFLUSH	R, T0, P1, P0, C2
	SHARED_INV	R, T0, P1, P0, C2
	PRIVATE_INV	R, T0, P1, P0, C2
TLB_INV	Multicast requests to all processors	
	ITLB	R, T0, V0, V1, V2, V3
	DTLB	R, T0, V0, V1, V2, V3
NRD_SB	Point to point request - R, T1, P1, P0, P2	
NRD_64	Point to point request to memory - R, T1, P1, P0	
NWR_SB	Point to point request - R, T1, P1, P0, P2, D16	
NWR_64	Point to point request - R, T1, P1, P0, D64	
NWR_HDR	Point to point request - R, T1	
SCRUB	Point to point request	
	TAG	R, T0, P1, P0
	SB	R, T0, P1, P0

Diagnostics & Utilities

8

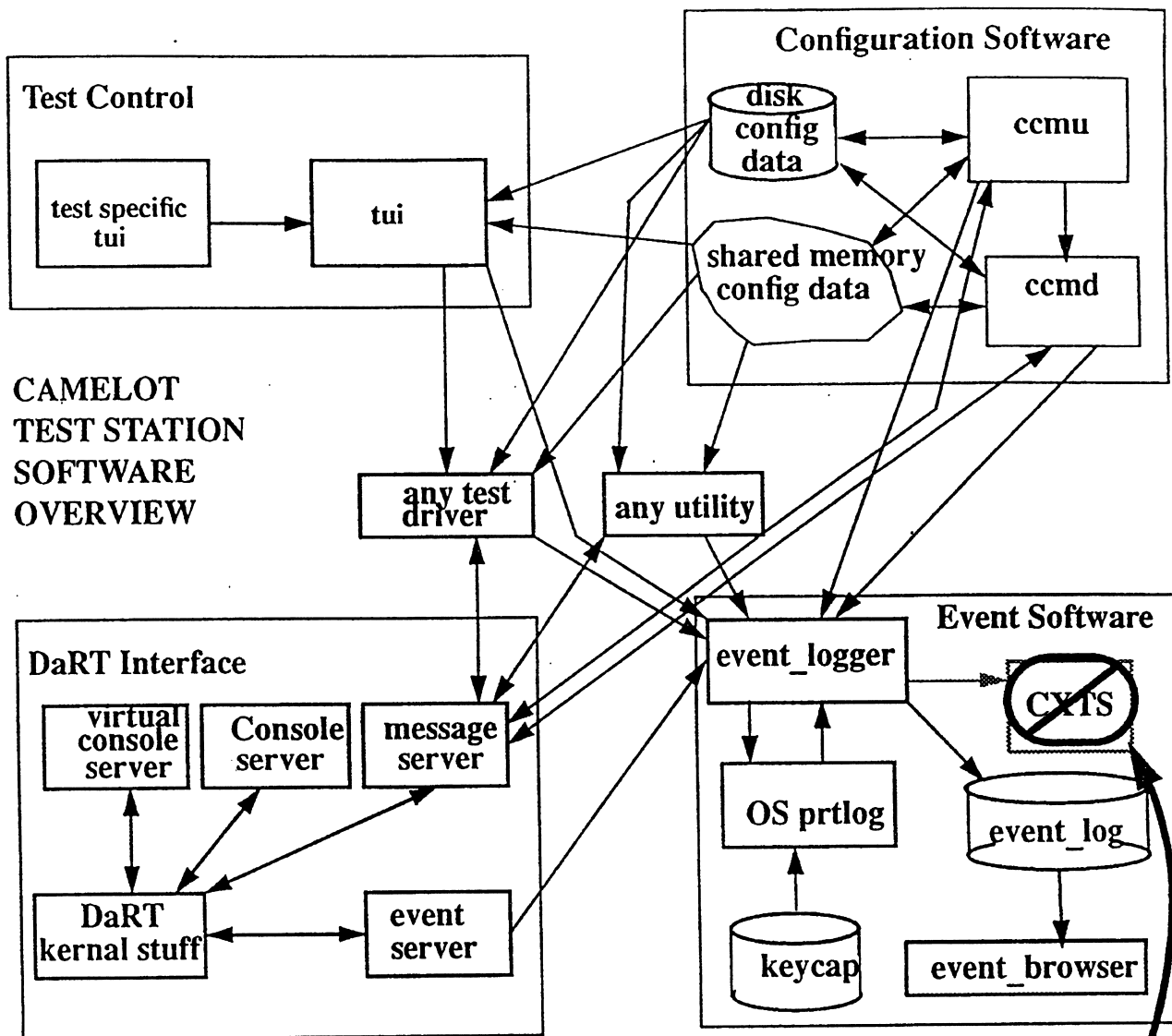
8.1 Test Station Introduction

8.1.1 Software

There are four distinct areas of software involved in Test Station Diagnostics and Utilities. These areas are:

- **Test Control** - The user interface to the software.
- **DaRT Interface** - The communication between the Test Station software and the SPP hardware.
- **Configuration software** - The physical hardware configuration data stores.
- **Event Software** - Event logging software

Figure 8-1
Test Station Software Overview

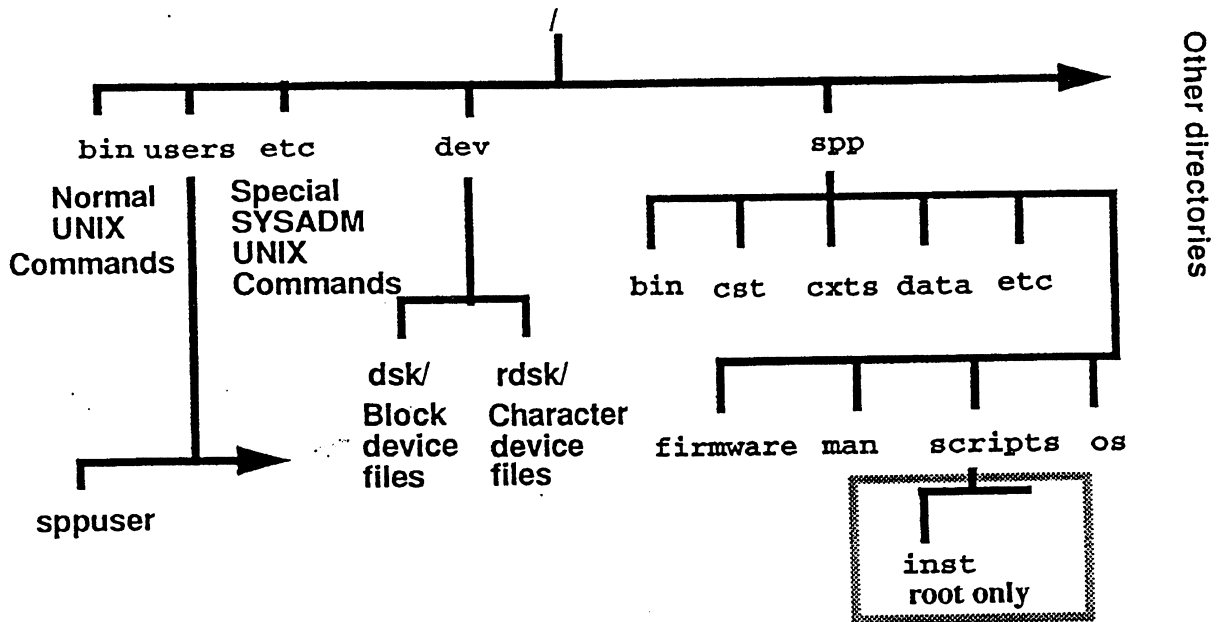


CAMELOT
TEST STATION
SOFTWARE
OVERVIEW

Has been replaced with CERS

C - onvex
E - vent
R - eport
S - ystem

8.1.2 The file System Tree



8.1.3 Important Test Station files

- /spp/bin
 - executables diagnostics and utilities
 - EX: sppring, cputest, event_logger
- /spp/data/
 - event_log
 - config/node_[#].cfg
 - config/config.s0123.c01234567.m128.obp
- /spp/firmware
 - OBP and MU firmware
 - backup firmware
- /spp/scripts
 - hard_logger, fix_hung_console and scripts used in trouble shooting
 - iod
 - inst/README
 - inst/ts.install
- /spp/os
 - sppboot - bootable file
- /spp/cxts
 - notification/notification.setup.default

8.2 Backup/Restore Commands

8.2.1 SPP-UX - System Files

- Backing up the complex's file systems from the test station:
 - Perform the backup as superuser on the SPP.
 - Setup a single CPU subcomplex using scm and start a xwindow on that subcomplex, to do so enter the following commands at the SPP-UX prompt:
 - `mpa -sc subcomplex_name xterm -display elmo_t:0 &`
- Use `fbackup` and `frecover` as described in the SPP-UX System Administration Guide.
- Use of the `dump` command does not require using a single cpu complex.

8.2.2 HP UX - Test Station Files

- **fbackup**
 - `/etc/fbackup -f /dev/rmt/3m -0 -i /`
 - The "-e" option can be used exclude directories.
- **Restore**
 - Perform as root.
 - Install backup tape in test station dat drive.
 - `/etc/frecover -f /dev/rmt/3m -i /`

8.2.3 Creating a Bootable Recovery Tape - HP-UX

- To make a tape the test station can boot off of:
 - Perform as root.
 - Install a blank tape in test station dat drive.
 - `/etc/mkrs -q -v -f /dev/rmt/3m -r /dev/rdisk/c201d6s0 -s -m 700`

8.2.4 Booting from the recovery tape:

- Turn on test station and hold down the [ESC] key.
 - At the prompt, type "search scsi" to find the scsi address of the tape drive.
 - The tape device will probably be at scsi address 3 and referred to as "scsi.3.0 HP HP35470A"
 - Enter "**boot scsi.3.0**" or use the scsi address which points to the tape drive.

8.2.5 Boot single on the Test Station

- Turn on the test station.
 - Press the [ESC] button and hold it down until you see the "ISL>" prompt.
 - To boot at the single init level, type: **boot -is**

8.5 Software/ Firmware Versions.

8.5.1 MU “core”

- The revision level of the MU “core” firmware that resides on the MU must be checked by reading an address on the MU as follows:
 - First enter `sppdsh` (if not already there), from the `sppuser` prompt type:
 - ◆ `sppdsh`
 - Then from the `sppdsh` prompt (`sppdsh$`) type:
 - ◆ `mget 0xf0807ffc`
 - A 32 bit word will be displayed to the screen instead of a version level, the following is an example of the current version:

```
sppdsh$ mget 0xf0807ffc
0x67fd2b05
```

- This hexadecimal word “0x67fd2b05” will represent the version level of the MU core currently installed on the MU board in NVRAM.
- To determine the version of the MU “core” on the test station, at the `sppuser` prompt enter:
 - `od -x /spp/firmware/mu_core | tail -2`
 - The last 4 bytes in the first line should be identical to what was read from the MU:

```
sppdsh$ od -x /spp/firmware/mu_core | tail -2
0077760 0000 0000 0000 0000 0000 0000 0000 67fd 2b05
0100000
```

8.5.2 Mu_firmware versions.

- To determine the version of mu_firmware that resides on the MU board in NVRAM, use the following command:
 - `/spp/bin/node_info`

EX:

```
elmo_d:/spp/os$ /spp/bin/node_info
Node Serial          UDP
ID  Number      IP Address    Base Port    MU Firmware
Version String
-----
0   1999770    130.169.128.0    675    "MU:CONVEX Version Info: 7.1"
      "PL:CONVEX Version Info: 7.0.1"
```

- There is also a software copy of mu_firmware that is kept on the test station in the `/spp/firmware` directory. The `diag_vers` command can be used to display it's revision level.
 - `diag_vers /spp/firmware/mu_firmware`

8.5.3 Diag_vers Command

The `diag_vers` command can be used to display the software versions of SPP test station executables. The following are a few examples.

- `event_logger`
 - `diag_vers /spp/bin/event_logger`
- `event_browser`
 - `diag_vers /spp/bin/event_browser`
- `ccmd`
 - `diag_vers /spp/bin/ccmd`
- `sppring`
 - `diag_vers /spp/bin/spring`

`Diag_vers` is usable on almost all diagnostics and spp software located on test station and can be executed in the `/spp/bin` director with an "*" as the argument to show revisions of all the executables located there.

8.5.4 SPP-UX Versions.

- At the SPP-UX OS prompt use the following command:
 - `vers /os/mach`
 - `vers /os/server`

8.6 CCMU Camelot Configuration Management Utility

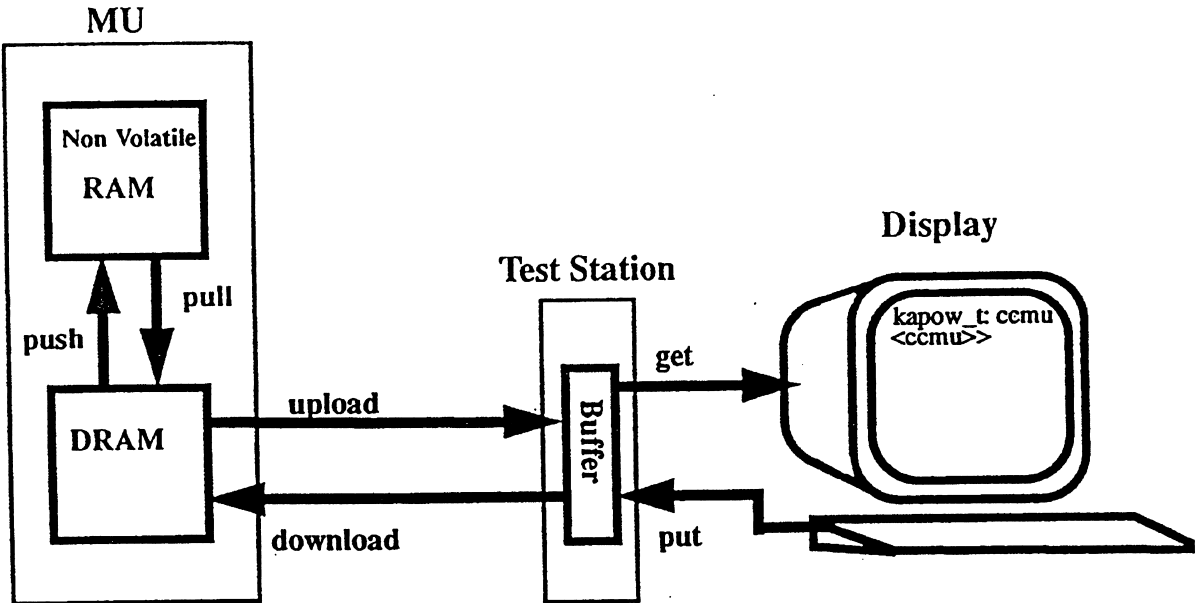
The ccmu utility is an interactive program that manipulates the hardware configuration database for the user.

- The database is actually generated and maintained by the ccmd daemon.
- The ccmu has a variety of commands dealing with all aspects of the hardware configuration.
- Commonly used commands:
 - **help** - print the help text
 - **quit** - terminate execution
 - **query** [**<node #> ...**] [**things**] - display all/specified thing
 - **select** [**<node #>...**] - select hardware
 - **clear** [**<node #>...**] - de-select hardware
 - **toggle** [**node#>...**] - toggle all/specified things
 - **auto** [**<node #>...**] - use the predetermined rules to set the config parms in a valid configuration for the present hardware on all/specified nodes.
 - **download** [**<node #>...**] - write parameters from the Test Station buffer to MU ram
 - **push** - write parameters from MU ram to the MU non-volatile memory
 - **pull** - read parameters from MU non-volatile memory to MU ram
 - **upload** [**<node #>...**] - read parameters from MU ram to the Test Station buffer

- **get** <item data name> - read parameters from the Test Station buffer and send them to the display
- **put** <item data name> <type> <new_value(s)> - write parameters from the display to the Test Station buffer
- **getstd** - get the standard / most often looked at parameters
- **get usememsize** - get the value of the memory size parameter
- **assign** <node #> <csb S/N> <complex S/N> <key - from TAC> - Set Complex S/N
##> Note ## this command causes a do_reset 1 of the node
- **assign** <node #> <csb serial number> - changes the address of the MU
Note ## this command causes a do_reset 1 of the node
- **copmod** <node #> <cop id> [-c] - enter various information for the specified cop.
- **cop** [<node #> [<<cop id>>]] - display the current information for a specific cop.
- **restore** <filename> - load parameters from a file.
- **save** <filename> - save parameters to a file

8.6.1 The CCMU Path Diagram

Figure 8-2
CCMU diagram



8.6.2 Important CCMU Parameters

- For more details on these parameters see Appendix F.
- The following was displayed with the `getstd` command.

```

elmo_d:/spp/bin$ ccmu
Welcome to the Convex Configuration Manager Utility
ccmu: 2.1 (Thu Dec 8 16:47:28 1994)
ccmu: defaulting to the following nodes: 0
ccmu> pull
ccmu> up
ccmu> getstd

node  value      parm  parm
  0   0x5882cf0f    1   INITWHAT
      STARTPL PLCPUINIT PLINITMEM PLLOADOBP INITMEM MUERRENABLE INITLAND
INITSLICE3 INITSLICE2 INITSLICE1 INITSLICE0 INITCXBAR DORESET PWRCLKCHECK GO
  0   0x330fffff    2   RUNWHAT
      MB1 MB0 XBAR_Q XBAR_S CCMC3 CCMC2 CCMC1 CCMC0 TCHIP0 TCHIP1 TCHIP2 TCHIP3
TCHIP4 TCHIP5 TCHIP6 TCHIP7 CPA3 CPA2 CPA1 CPA0 IO1 IO0 MAUI MU
  0   0x00000000    3   NODEID
  0   0xffffffff    4   ASSERTRESET
  0   0x030fffff    5   DEASSERTRESET
      XBAR_Q XBAR_S CCMC3 CCMC2 CCMC1 CCMC0 TCHIP7 TCHIP6 TCHIP5 TCHIP4 TCHIP3
TCHIP2 TCHIP1 TCHIP0 CPA3 CPA2 CPA1 CPA0 IO1 IO0 MAUI MU
  0   0x00000002   17   USEMEMSIZE
  0   0x00000004   18   USEBANKFIELD
  0   0x00000000   19   USECACHESIZE
  0   0x0000ffff   218  TRACEMU
  0   0x00000080   219  STATUSMU
  0   0x00007cc7   220  APPLY_POWER
      IO VEE_CLK VHE_HP VHC_HP VCC_HP VCC_MB1 VCC_MB0 VTT VDL VDD_GA
  0   0x00000000   256  MU_CONFIG
  0   0x800f13ff   221  AVAIL_HW
      NODE SLICE3 SLICE2 SLICE1 SLICE0 IO MB1 MB0 TCHIP7 TCHIP6 TCHIP5 TCHIP4
TCHIP3 TCHIP2 TCHIP1 TCHIP0
  0   0x00000001   258  AVAIL_NODES
  0   0x00000001   257  PRESENT_NODES
  0   0x0000ff01   259  HW_CONFIG
      MEM_3_BANK_1 MEM_3_BANK_0 MEM_2_BANK_1 MEM_2_BANK_0 MEM_1_BANK_1
MEM_1_BANK_0 MEM_0_BANK_1 MEM_0_BANK_0 CMC_DARK
  0   0x000f4240   222  CLOCK_10MSEC
  0   0x00000000   223  APPLY_CLOCK
  0   0x00000000   224  INFO_MU
  0   0x000007cf   241  CPA_ERR_ENABLE
  0   0x00000000   242  CPA_PMON_LATENCY
  0   0x00000002   243  CPA_PMON_CONTROL
  0   0xfc7f3fff   244  CMC_ERR_EN0
  0   0x00000000   245  CMC_ERR_EN1

```

8.7 Configuration Errors

If you are not sure if the system contains the correct configuration of the hardware, the following procedure should give you a clean and correct configuration.

8.7.1 Correct From Scratch - Using CCMU

- Power on the Node.
- Wait 30 seconds to allow the regen command to complete.
- Invoke ccmu.
- Type up.
- Type select n (or the Node number you want).
- Type auto (to generate a configuration based on the hardware recognized by the regen command).
- Type down (to download the configuration in to the MU DRAM).
- Type push (to push the configuration into the MU NVRAM).

8.7.2 Correct From File

- cd /spp/data/config
- invoke ccmu
- type restore [file name of previously made file using the save command]
- type down
- type push
- type quit to exit

8.7.3 Using Saveall and Restall

Configuration errors can also be corrected by using the “restall” script.

- Saveall is a script run from the sppuser prompt that completely backups up all of nvram, firmware as well as parameters.
 - It creates 3 files in the PWD.
 - Will be explained in greater detail in section 8.23 on page 196.
- Restall is saveall’s counterpart.
 - Used to restore the files created with saveall.
 - Will not work with parameter only files created in CCMU.
 - See section 8.24 on page 197 for more details.

8.8 Open Boot Program

8.8.1 OBP Open Boot Prom Architecture

OBP (secondary loader) is a software architecture for the firmware that controls a computer before the operating system has begun execution. It runs entirely in the CPU(s) of the computer - not in the MU or the test station.

- It performs three functions:
 - Auto configuration: it is designed to completely auto configure the I/O subsystem and to provide to the OS a complete and ACCURATE description of all the hardware available to the OS.
 - Boot the OS: The simplest but essential function of OBP.
 - Configuration Parameters:
 - ◆ Low-Level HW configuration.
 - ◆ OS Boot configuration parameters.
 - ◆ Mach and Unix configuration parameters

8.8.2 OBP Commands


- Often used OBP commands:
 - **reset** - Reboots the obp, use after modifications.
 - **printenv** - Display all parameters.
 - **setenv boot-device** - Specifies and sets a new boot device.
 - **setenv boot-directory** - Specifies and sets a new directory where the boot files are found.
 - **show-devs** - Displays all devices known to the OBP.
 - **devalias** - Displays all aliases stored in nvram.
 - **nvalias sd0a /mbus@0,ffec0000/sbus@f,fcffff00 Convex,afws@1,10000/sd@2,0:a** - Creates an aliase in nvram telling the system to boot the OS from partition "a" on the disk drive physically located at scsi id2, sbus slot 0-unit 0 on siop0.
 - **mkmap -n 0 /mbus@0,ffed0000/sbus@f,fcffff00/CRES,cddi@1,400000** - Mkmap is used to link all networking devices and tape drives to the kernal, otherwise these devices will not be seen by the system.
 - **boot** - Boot SPP-UX.

8.8.3 Example of a normal OBP banner.

```
OBP Hard boot
Exemplar SPP1000/XA, OBP Release 1.0.0.0 Version 1 created 94/06/2
4 CPUs, 512 MB memory installed, 0 MB network cache, Landmarc I/O
Complex Serial Number: 65536, Node Serial Number: 1999027.
Network address 0:0:0:0:0:0, Host ID #: 00000000.
```

```
This version of OBP supports loadhack.
Type help for more information
[0:2] ok
```

Node# : Proc#



8.8.4 OBP standard parameters - defaults are GSM

```
[0:2] ok printenv
```

Parameter Name	Value	Default Value
log-device	/console	/console
output-device	/console	/console
input-device	/console	/console
lu-hack?	true	true
u0sb0	afws-fcode	
u0sb1		
u0sb2		
u0sb3		
u1sb0		
u1sb1		
u1sb2		
u1sb3		
sbus-probe-list	U1SB3 U1SB2 U1SB1 U1SB0 U0SB3 U0SB2 U0SB1	
U0SB0	U1SB3 U1SB2 U1SB1 U1SB0 U0SB3 U0SB2 U0SB1	
U0SB0		
keyboard-click?	false	false
fcode-debug?	false	false
local-mac-address?	false	false
tz	CST6CDT,91/0300,301/0100	CST6CDT,91/0300,301/0100
root_fs_node	0	0
pm_node_count	1	1
incarnation	1	1
boot_node_list	0	0
accept_procs	0	0
diag-device	none	none
diag-file	none	none
boot-device	sd0a	sd0a
boot-directory	/os	/os
tunables-file	tunables	tunables

server-file	server	server
boot-file	mach	mach
boot-args		
ramdisk-file	ramdisk	ramdisk
pvmcti-memory-size	0	0
cluster-boot?	false	false
load-tunables?	true	true
load-server?	true	true
load-ramdisk?	false	false
auto-boot?	false	false
watchdog-reboot?	false	false
screen-#columns	80	80
screen-#rows	24	24
swdump?	true	true
panic_query?	false	false
do_kernelDump?	true	true
do_serverdump?	true	true
panic_gracefully?	true	true
do_crashreport?	true	true
panic_reboot_option	none	none
use-nvramrc?	true	true
nvramrc		
security-mode	none	none
security-password		
security-#badlogins	0	<no default>
last-hardware-update		<no default>
mfg-switch?	false	false
diag-switch?	false	false

8.8.5 OBP PVM-CTI VS GSM

- If running in PVM-CTI mode there will be an OBP window for every node available.
 - cluster-boot - will be set to true
 - pvmcti-memory-size - must be set
- If running in GSM mode there will be only one OBP window for the system console.
 - the above defaults are for GSM
 - cluster-boot - will be set to false

8.8.6 Turning off OBP

It is currently necessary to turn off OBP when running diagnostics due to some sharing of registers on the MU. Failure to do so can result in misleading diagnostic failures.

- To turn off the OBP:
 - Enter CCMU.
 - Issue a "pull" and then an "UP".
 - Then "put" the value of INITWHAT in order to turn off the PLLOADOBP (bit 23).
 - EX:

```
elmo_d:/users/sppuser$ ccmu
Welcome to the Convex Configuration Manager Utility

ccmu: 2.1 (Thu Dec 8 16:47:28 1994)
ccmu: defaulting to the following nodes: 0

ccmu> pull
ccmu> up
ccmu> get initwhat
node  value      parm  parm
  0   0x5880cf0f   1    INITWHAT
      STARTPL PLCPUINIT PLINITMEM PLLOADOBP MURRENABLE INITLAND INITSlice3
      INITSlice2 INITSlice1 INITSlice0 INITCXBAR DORESET PWRCLKCHECK GO

ccmu> put initwhat 0x5800cf0f
ccmu> down
ccmu> push
ccmu> exit
elmo_d: do_reset
```

8.9 Xconfig All

XConfig is a GUI Hardware configuration utility

Figure 8-3
Xconfig Display

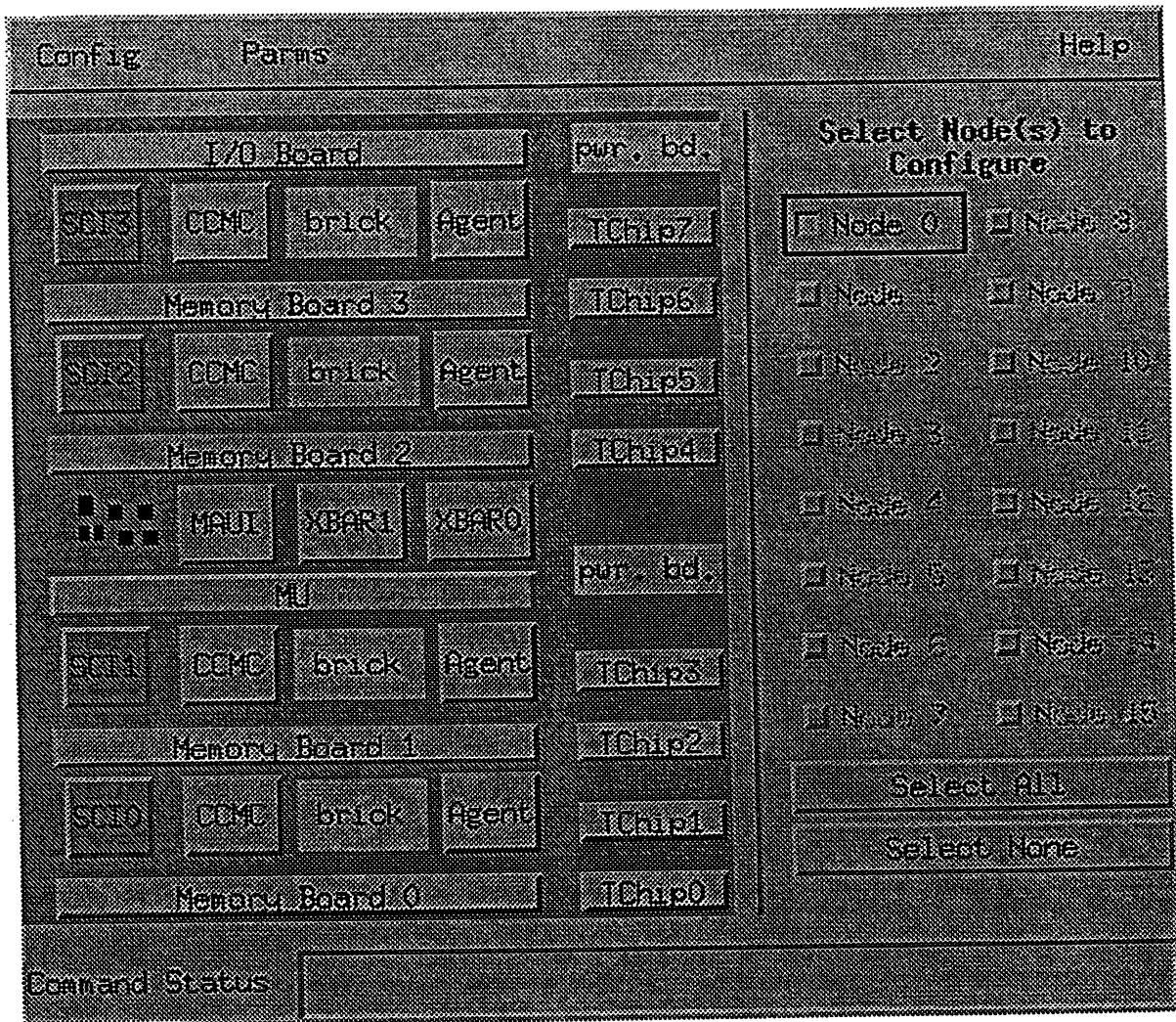


Figure 8-4
Xconfig Config Pop-up Menu

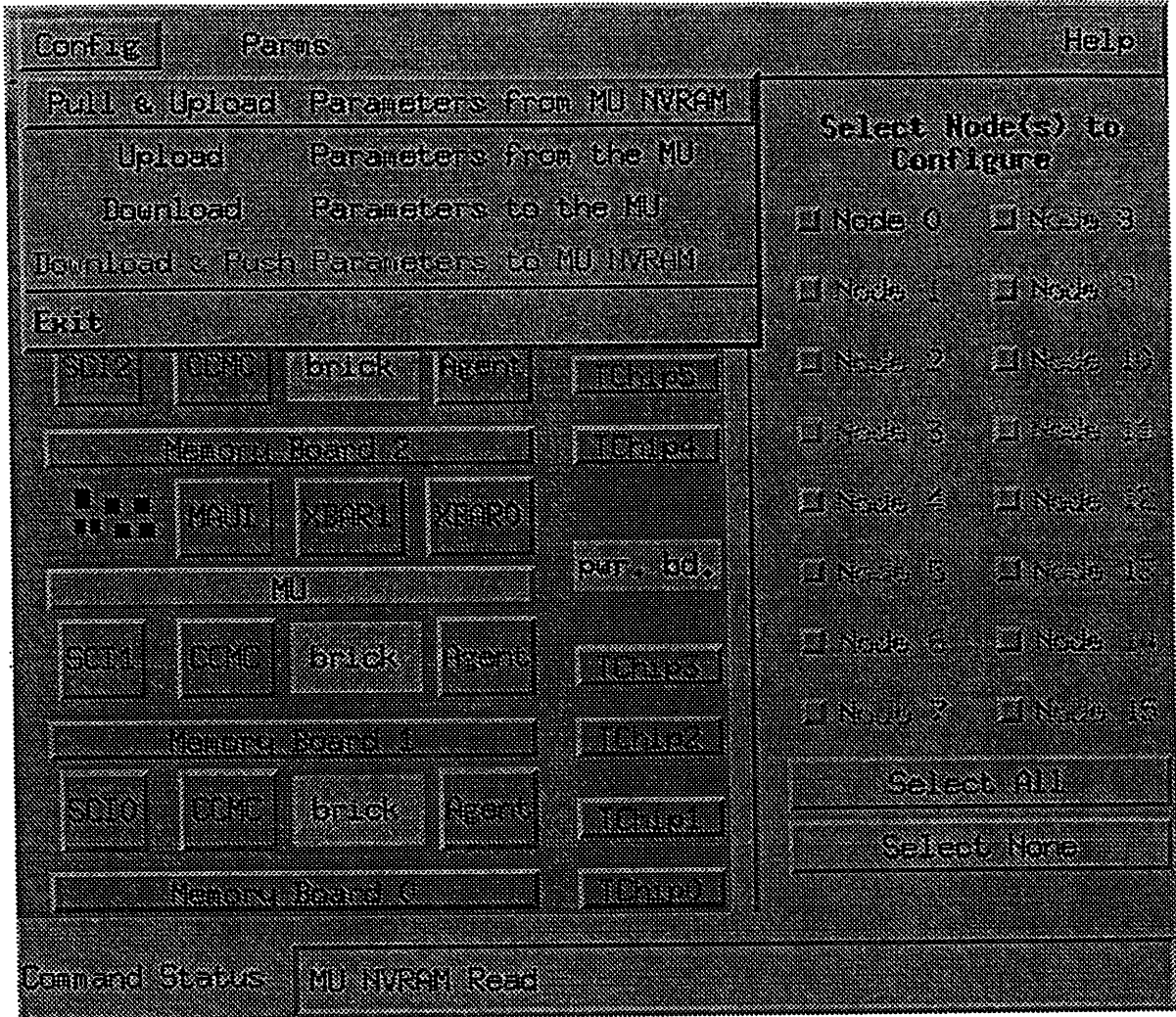


Figure 8-6
Xconfig Parm/Clocks Pop-up Menu

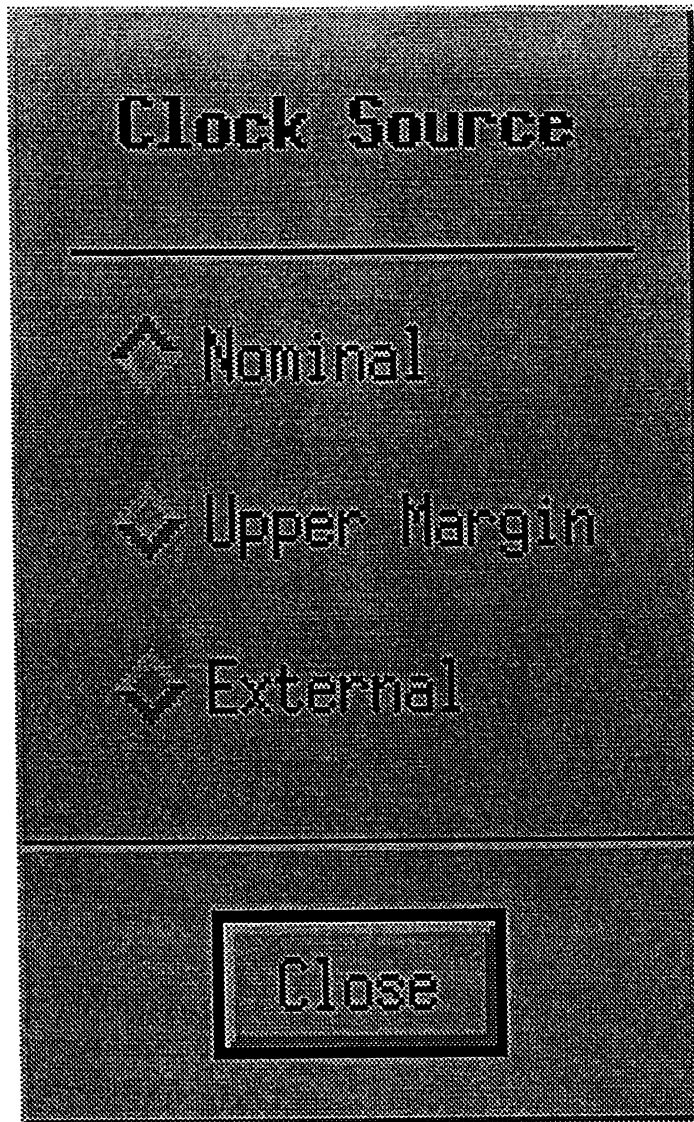
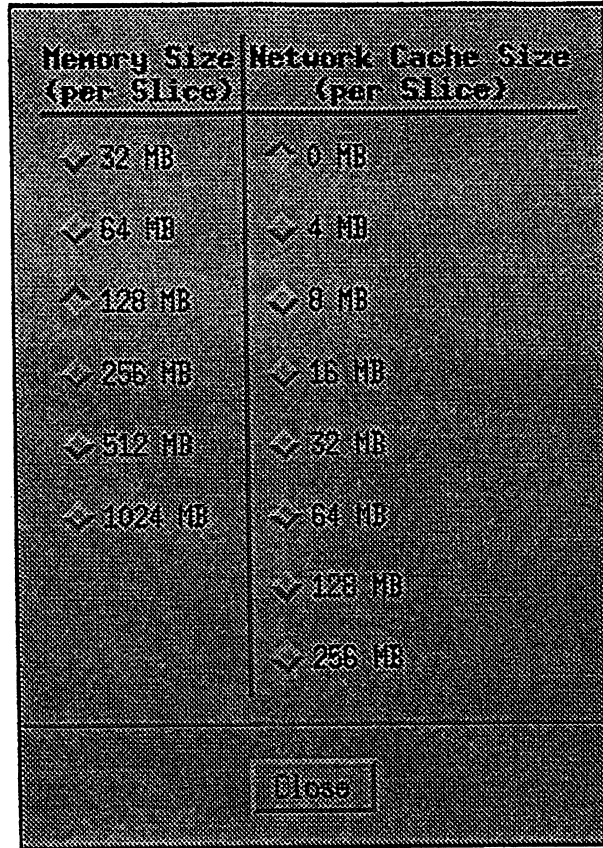


Figure 8-7
Xconfig Parm/ Memory Options Pop-up Window



The image shows a pop-up window with two columns of options. The left column is titled 'Memory Size (per Slice)' and the right column is titled 'Network Cache Size (per Slice)'. Both columns have a list of values with a small diamond-shaped cursor next to each. At the bottom center of the window is an 'Erase' button.

Memory Size (per Slice)	Network Cache Size (per Slice)
32 MB	0 MB
64 MB	4 MB
128 MB	8 MB
256 MB	16 MB
512 MB	32 MB
1024 MB	64 MB
	128 MB
	256 MB

Erase

Figure 8-8
Xconfig Help Pop-up Window

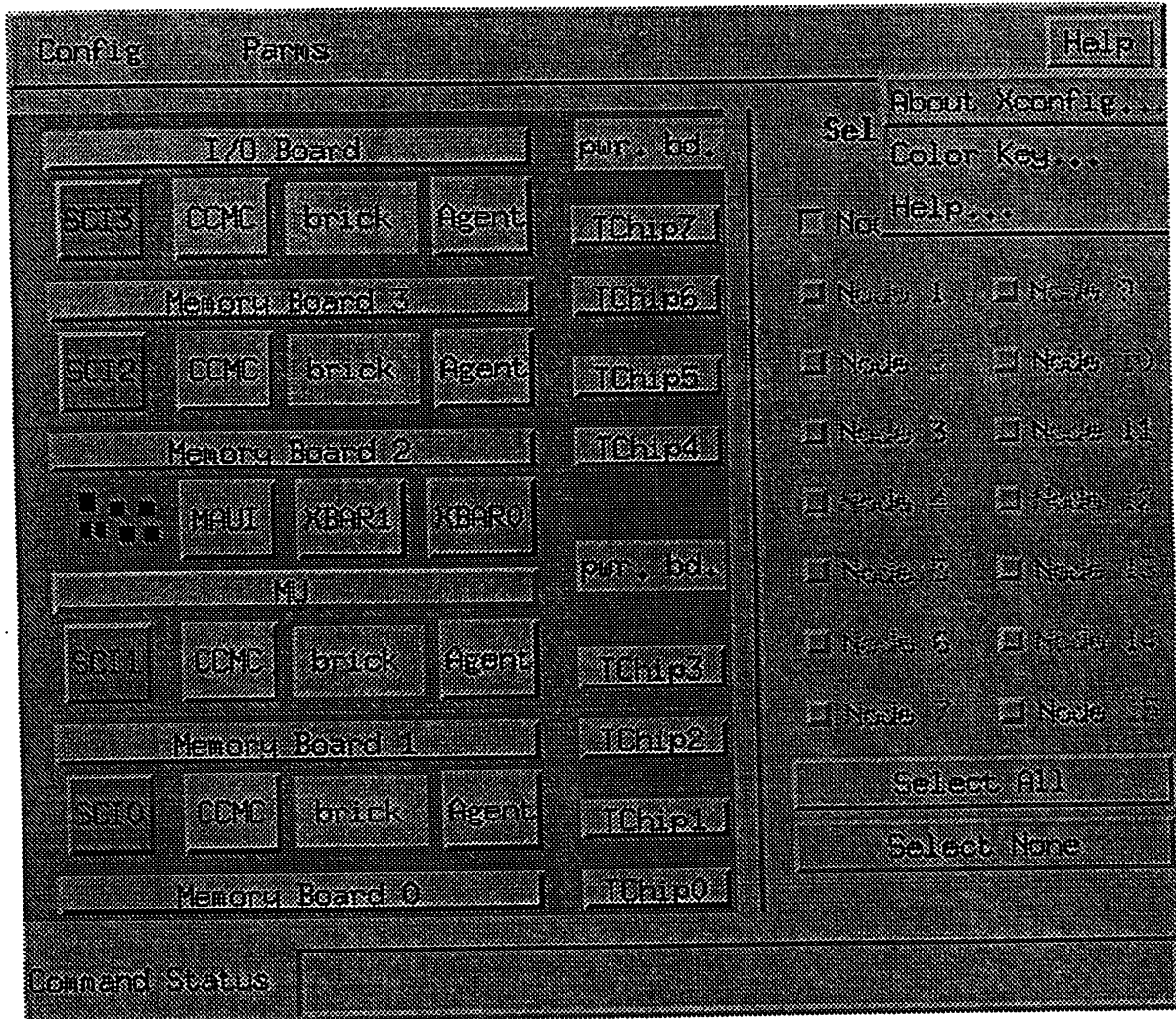
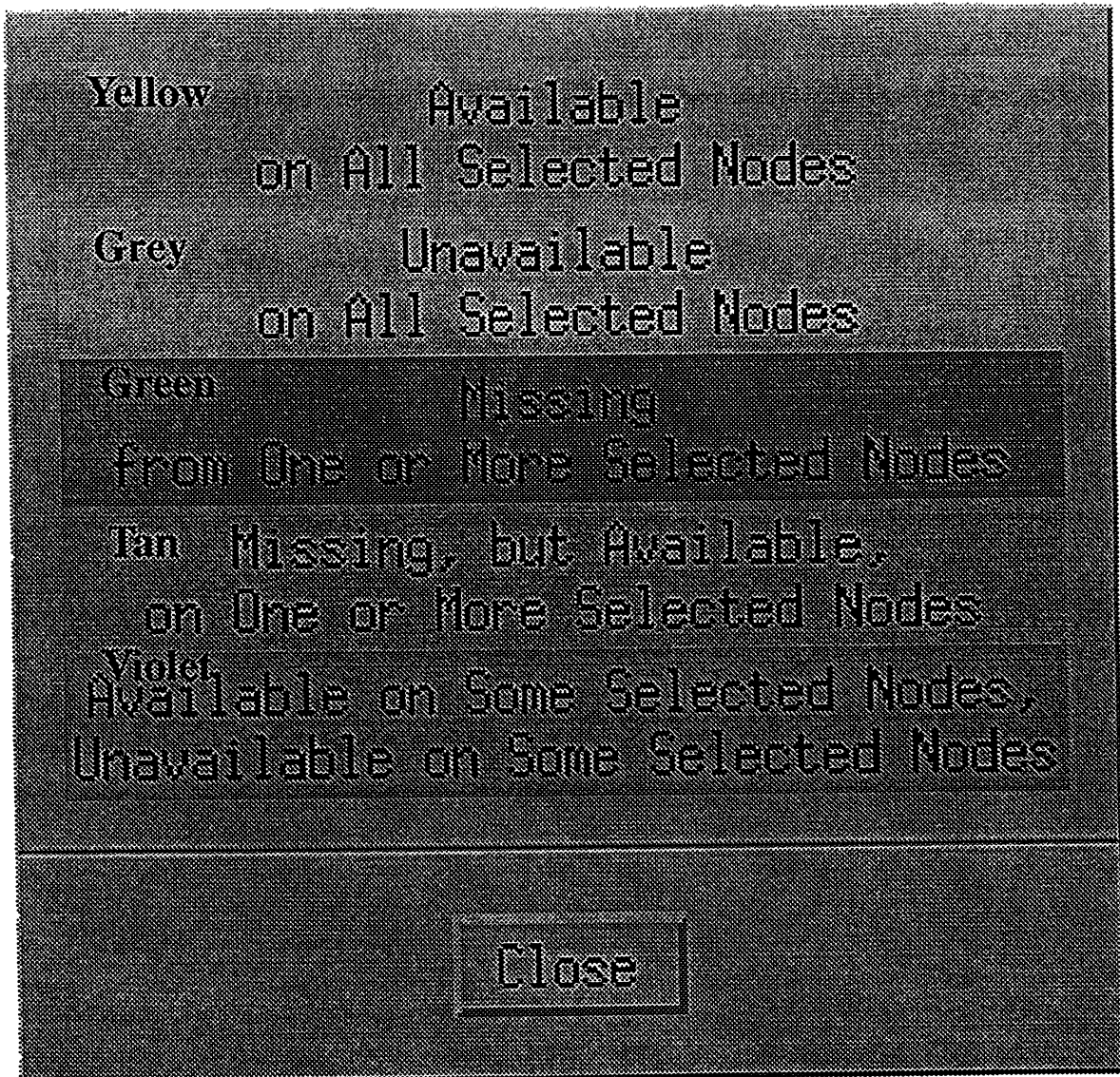


Figure 8-9
Xconfig Help/Color Key Pop-up Window



8.10 Cputest: T-CHIP Diagnostic

The cpu diagnostic test “Cputest”, is provided by HP.

- The following is an example (SPP-1000) of how to run it:

```
spp1_t% /spp/bin/cputest
```

```
Starting cputest at Thu Dec 14 14:34:37 1995
cputest:  1 node available:  0
cputest:  8 cpus available on node 0:  0 1 2 3 4 5 6 7
cputest:  estimated execution time is 2:24 (hours:minutes)
          Subtest 1000 - Processor Module Subtest
```

- This is a Pass/Fail test, no updates will occur while the test is running.
- It takes 18 minutes per processor on the SPP-1000.
- On the SPP-1200, the test takes about 22mins no matter how many processors are installed.
- The test runs in parallel across all nodes that are enabled, testing multiple nodes will not increase test time.
- Nodes and/or cpus can be deconfigured using xconfig.
- Failures:
 - failures are usually time-outs
 - ◆ timeout for the SPP1200 is about 30mins
 - some data cache parity errors
 - and HPMCs

8.11 CST - Convex Scan Test

Warning: Invoking CST while SPP-UX is running will cause a crash.

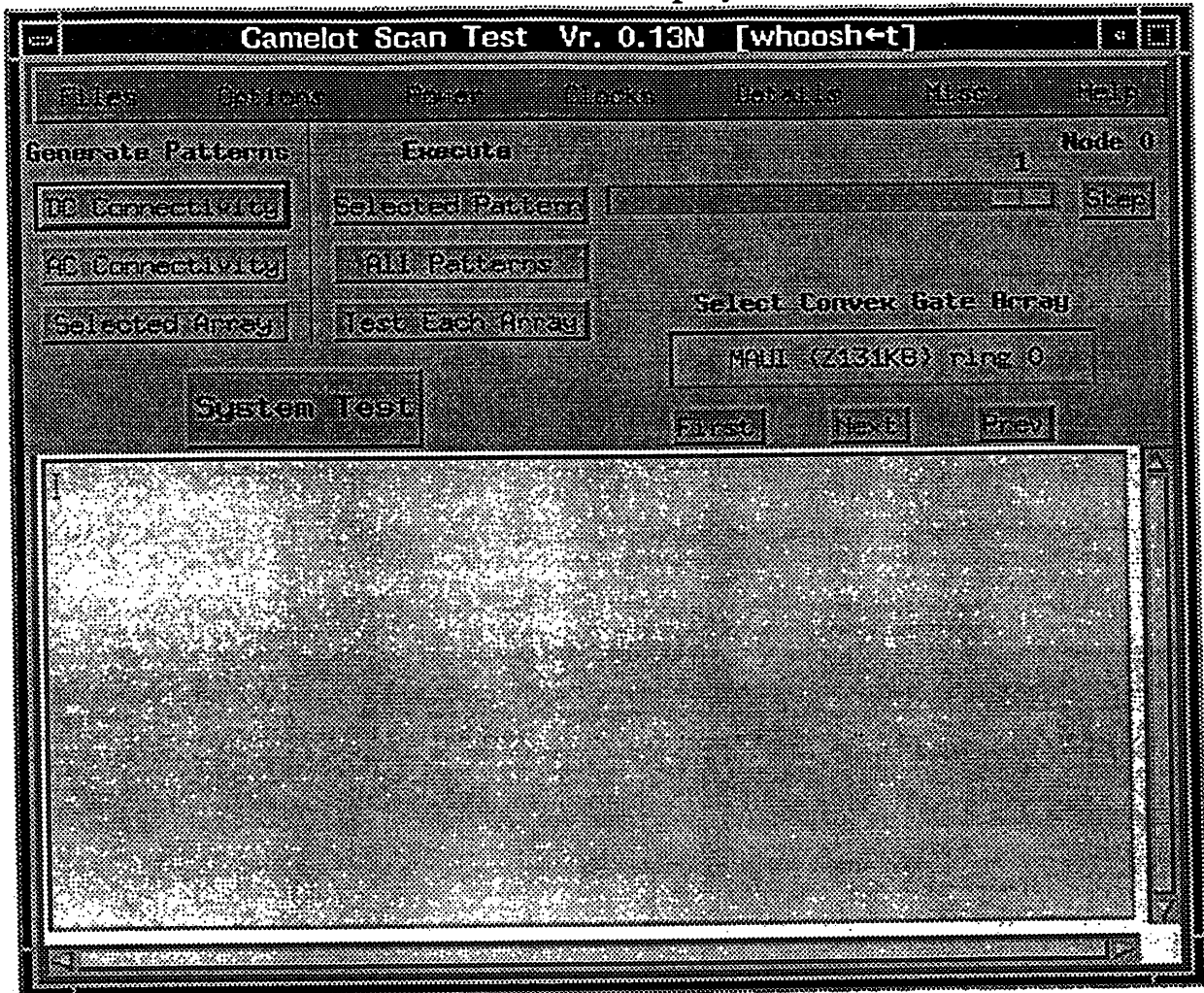
- Options:
 - -V dumb terminal mode (text mode).
 - - N demo mode (No Hardware calls), SAFE to run if SPP-UX is running.
 - -v returns the version, SAFE to run if SPP-UX is running.
 - Argument "Node_number".
 - Logs output to /spp/cst/cst.log.
 - ◆ Will make one backup file.

Note: use "resize" to fix screen after exiting CST -V

Note: *If run on SPP-1200 CST scans the T'-Chips. A defective T'-Chip can cause erroneous error messages.*

WARNING: A "pull" from nvram must be executed after running CST. Failure to do so will cause the system not to boot or possible fail intermittently.

Figure 8-10
CST GUI Display



- Point & click operation
- AC/DC Internal & External connectivity
- Standard test controls:
 - Limit
 - No limit
 - loop
 - pause on failure

Figure 8-11
CST Scan Ring Partitioning

6 Individual Scan Rings Per Node

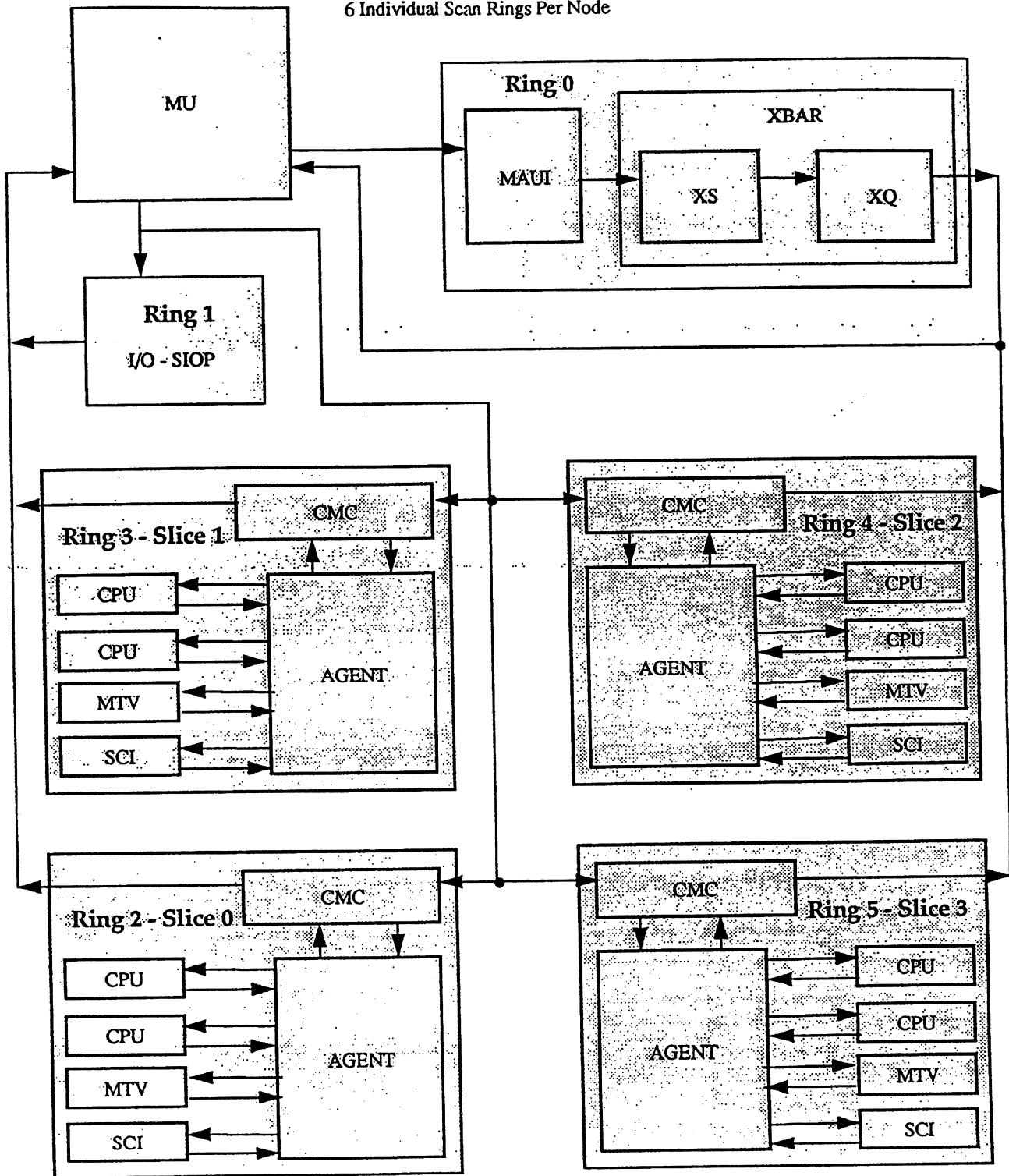


Figure 8-12
CST Files Menu

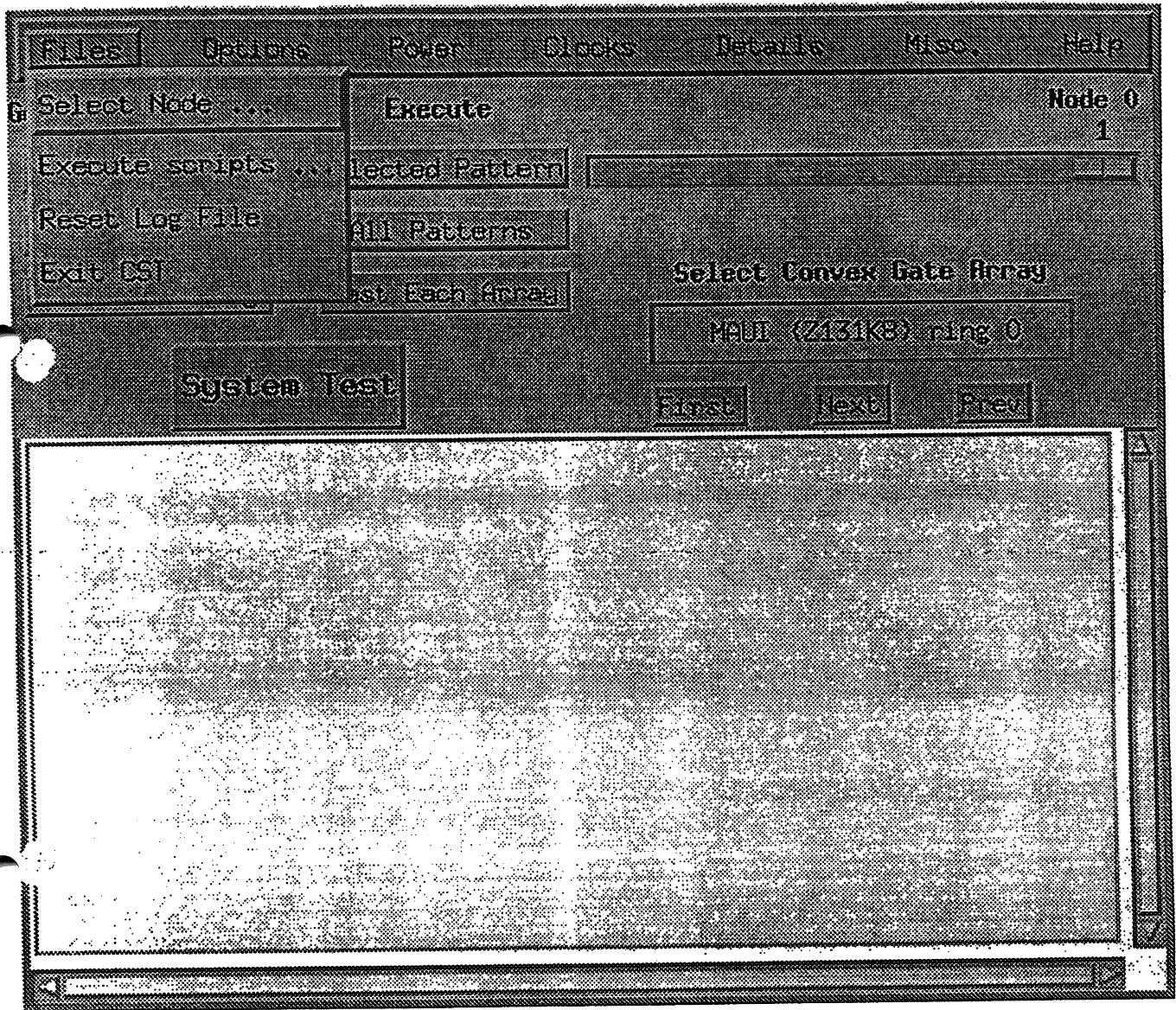


Figure 8-13
CST Options Menu



Figure 8-14
CST Power Menu

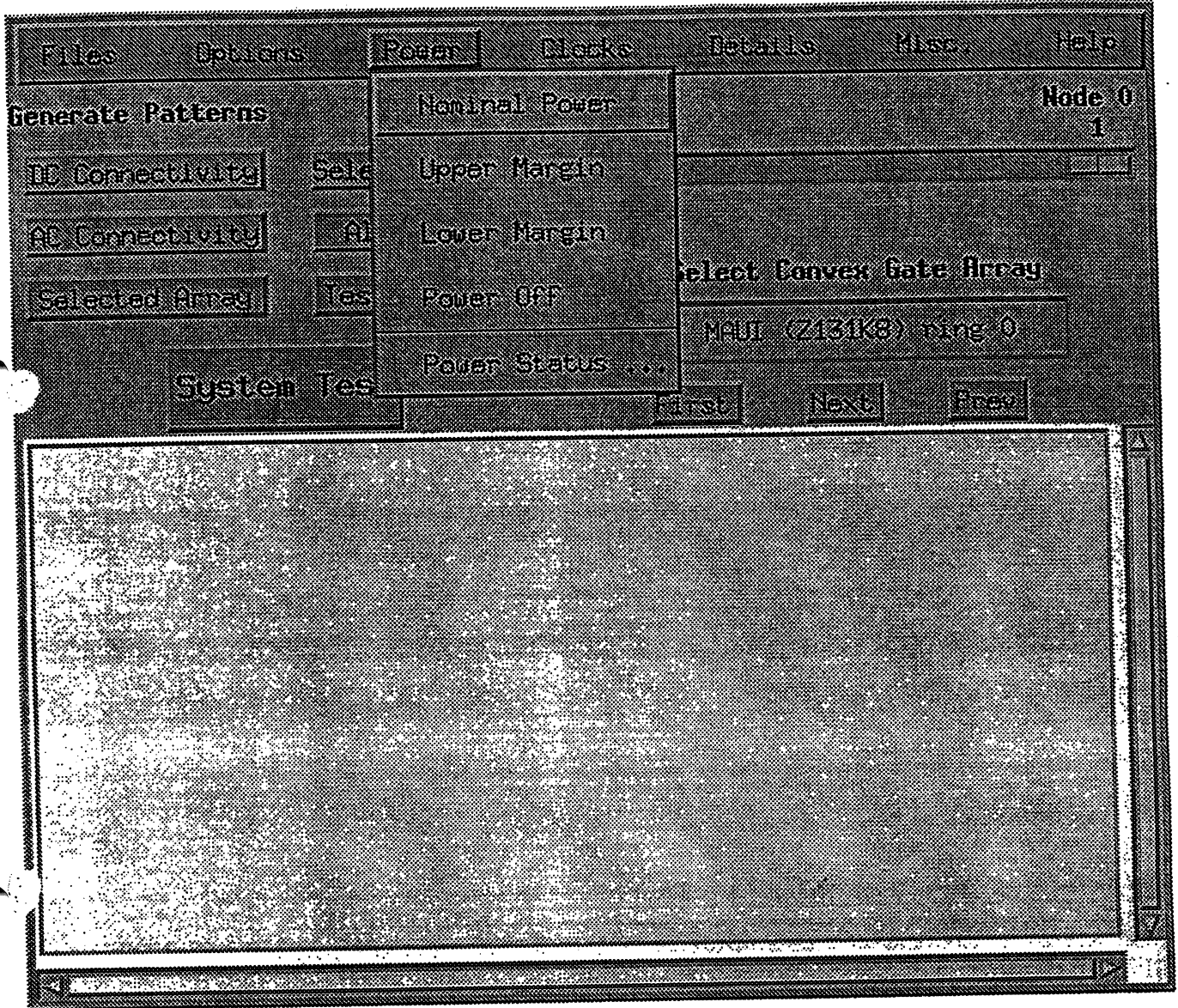


Figure 8-15
CST Clocks Menu

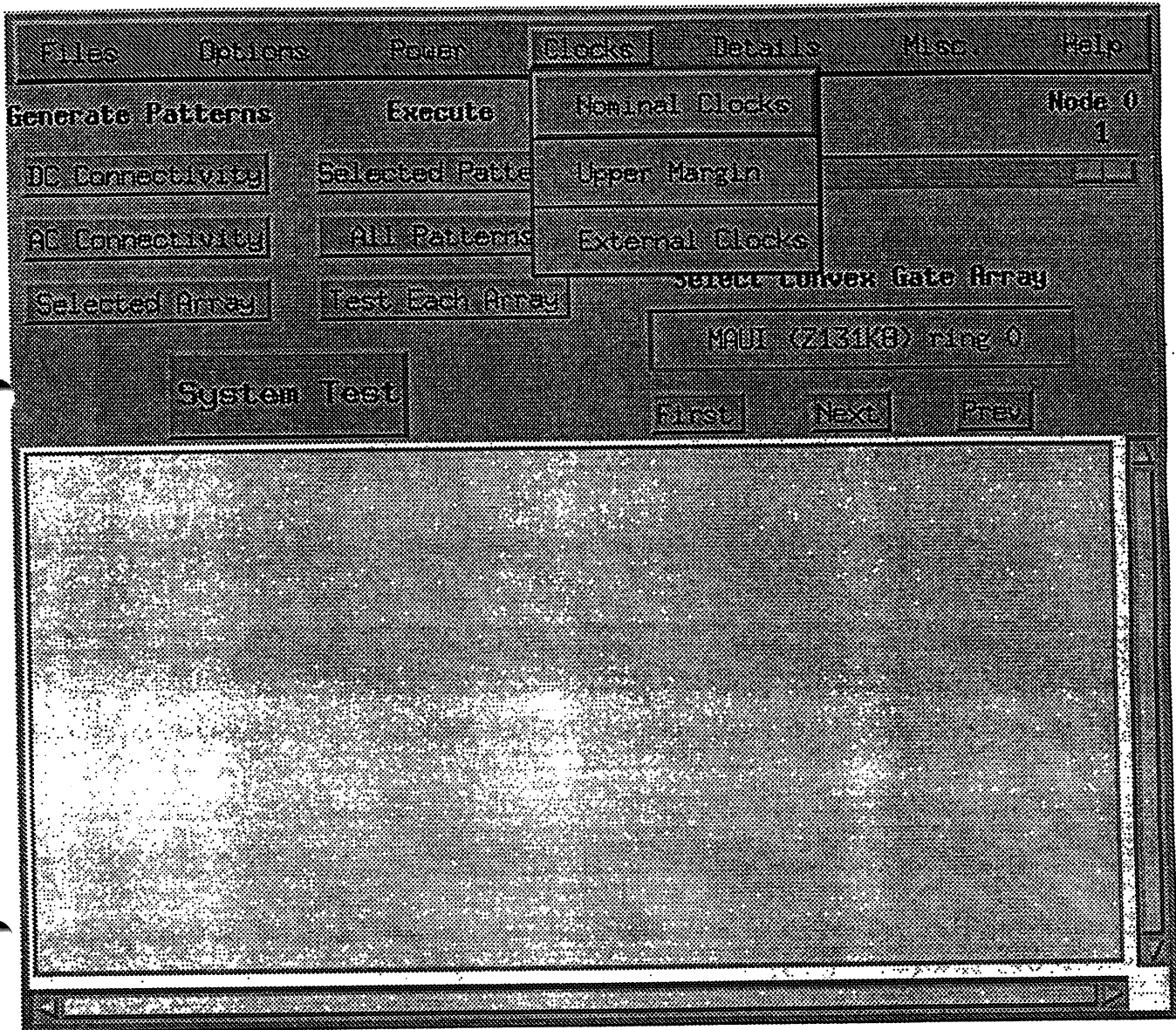


Figure 8-16
CST Details Menu

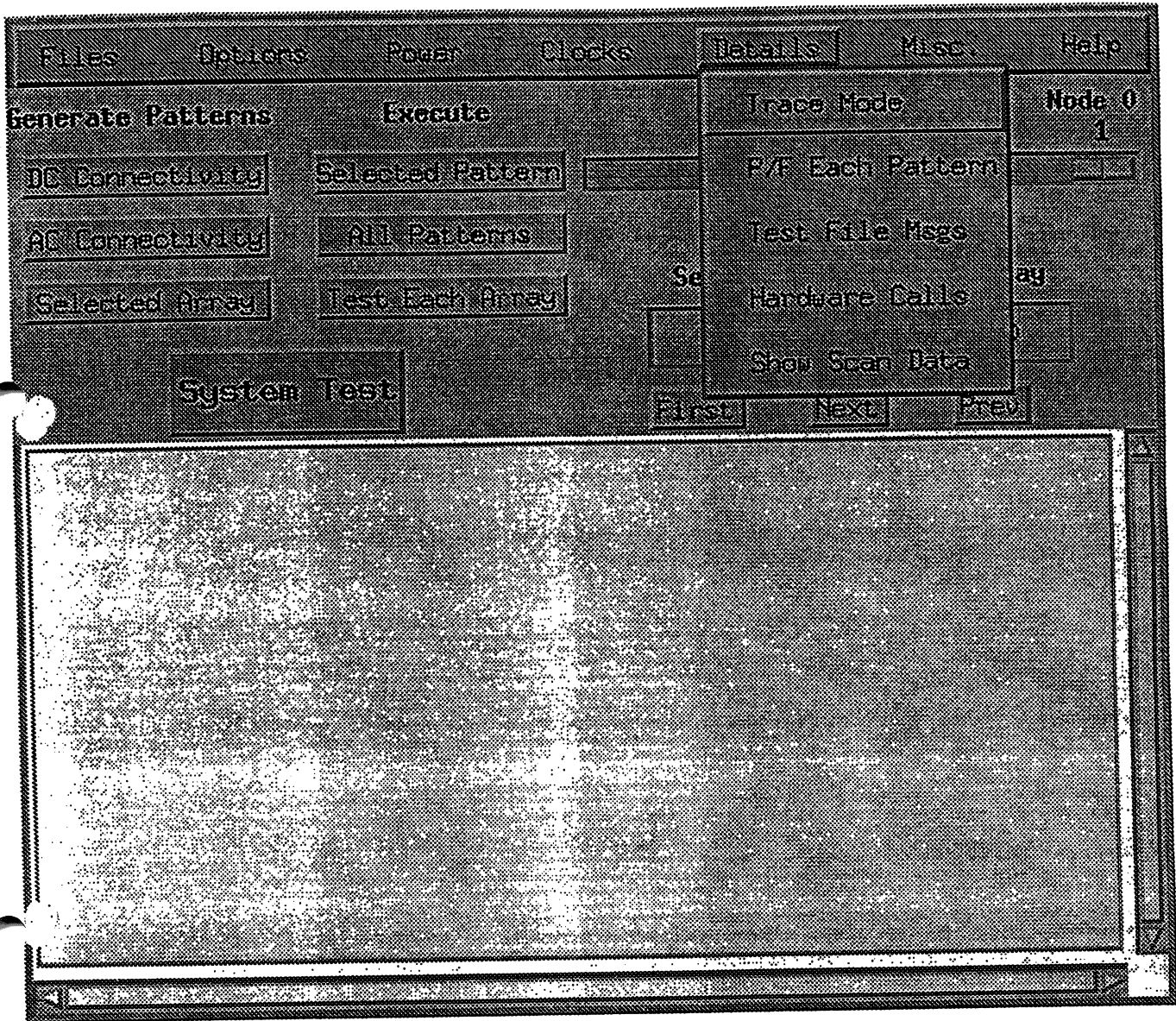


Figure 8-17
CST Misc. Menu

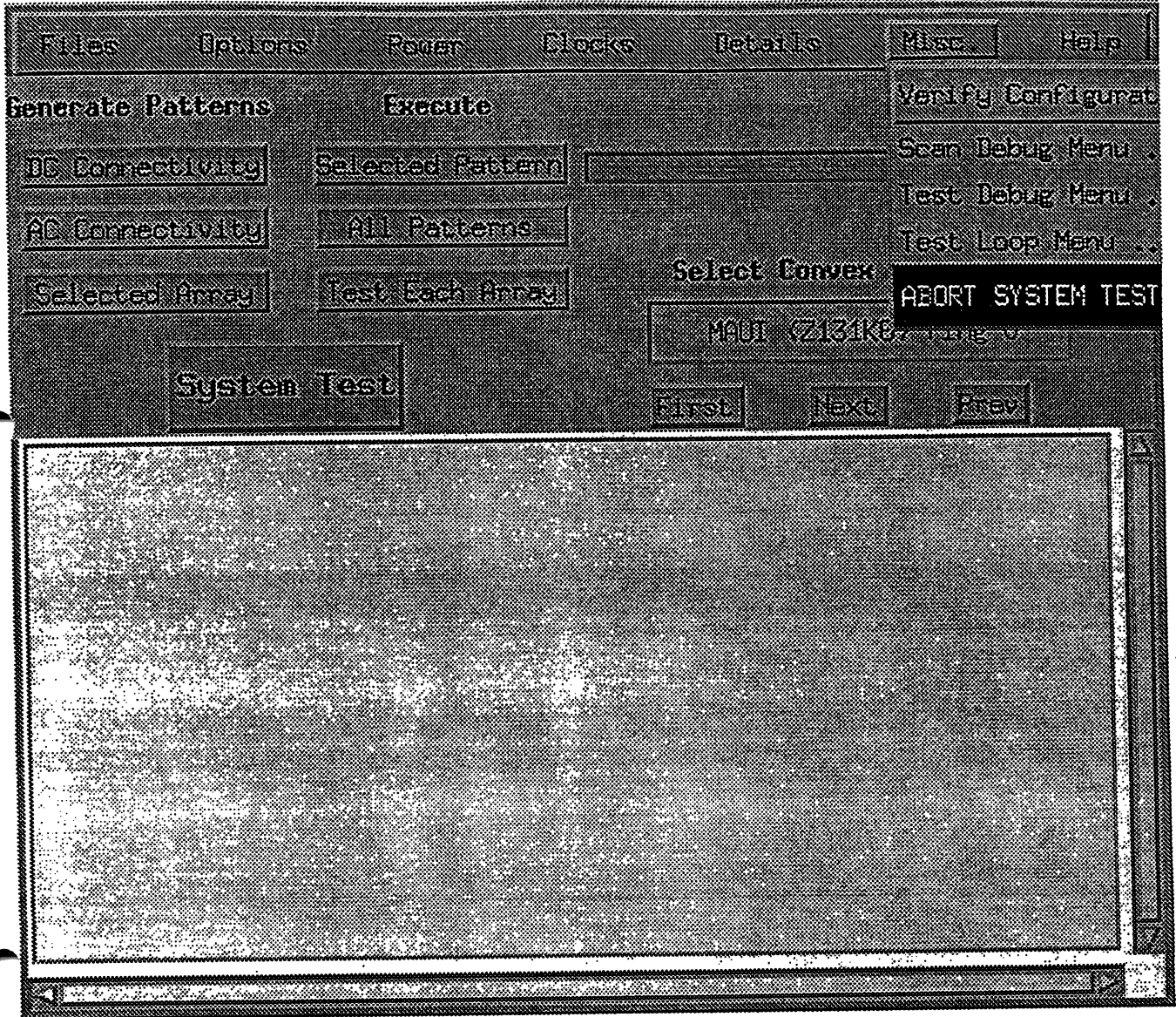
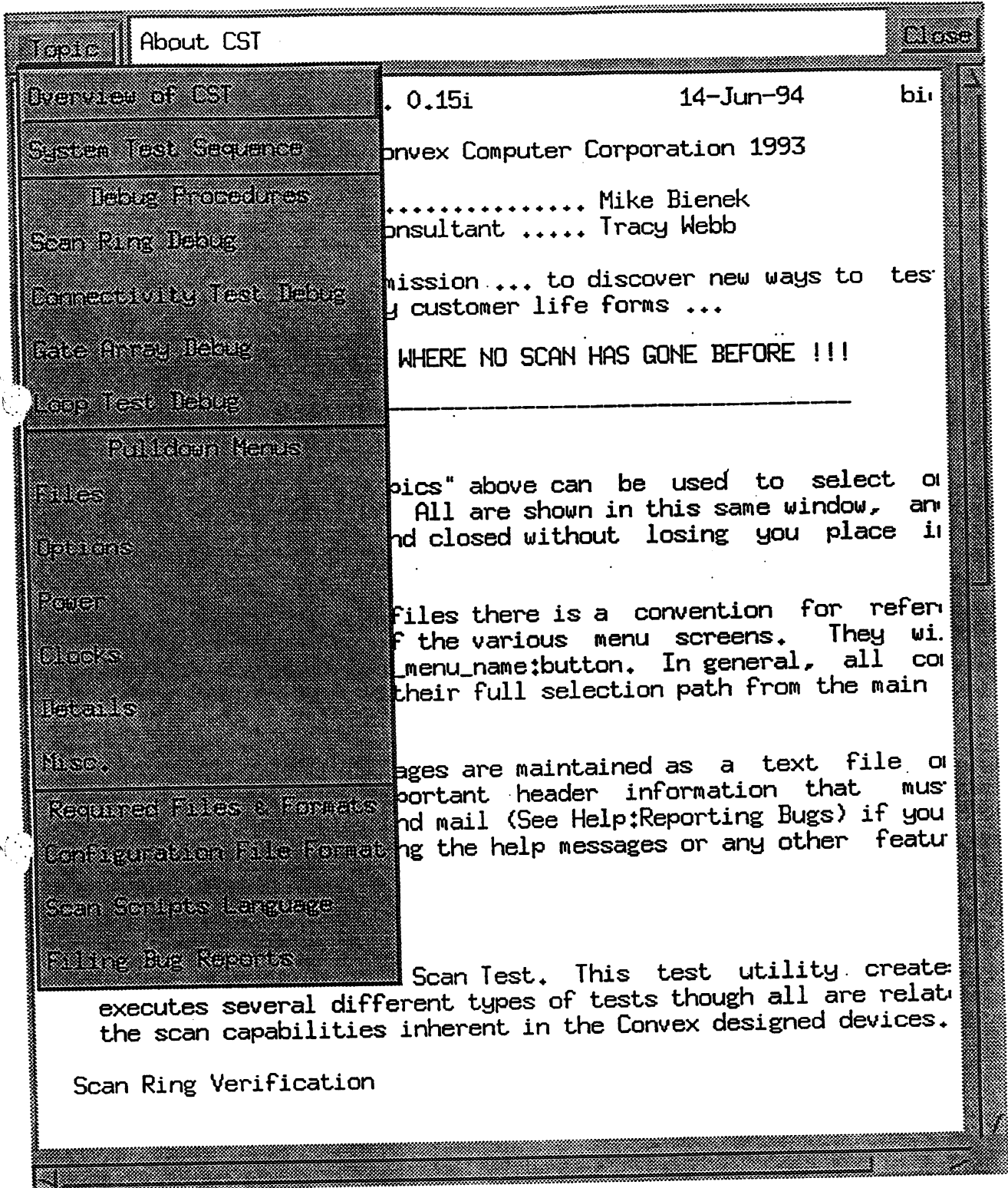


Figure 8-18
CST Help Screen



8.11.1 CST Connectivity Failure Message

Upper clocks selected
Limit Patterns ... Off
Limit Errors Off
Details on Fail .. On
Stop On Error Off

Begin System Test ...

Selecting NOMINAL Power by default
Available scan rings ... 0 2 3 4 5
Ring configuration test PASSED
ByPass Scan Path PASSED
Boundary Scan Path PASSED
Internal Scan Path PASSED

> Generating DC Patterns

Wire MCMB0MECC4 (ring 2 -> ring 2)
Driver: CCMC2 Z131C0.AK31 CM_MC_MECC_4 bit 149 Hi
Recver: BPR Z045B9.12 PDAT<2> bit 58 Lo

Pattern 0001 - dc_pat.001 FAILED 1 error

Wire MCMB0MECC4 (ring 2 -> ring 2)
Driver: CCMC2 Z131C0.AK31 CM_MC_MECC_4 bit 149 Hi
Recver:-- BPR Z045B9.12 at.00 2)
Driver: CC Hi
Recver: BP DAT<2> bit 58 Lo

Pattern 0007 - dc_por

1C0.AK31 CM_MC_MECC_4 bit 149 Hi
R PDAT<2> bit 58 Lo

Pattern 0008 - dc_pat.008 FAILED 1 error

Wire MCMB0MECC Recver: BPR Z045B9.12

PDA-T009 - dc_pat.009 FAILED 1 error

Wire MCMB0MECC4 (ring 2 -> ring 2)

Driver: CCMC2 Z131C0.AK31 CM_MC_MECC_4 bit 149 Hi
Recver: BPR Z045B9.12 PDAT<2> bit 58 Lo

Pattern 0009 - dc_pat.009 FAILED 1 error

Wire MCMB0MECC4 (ring 2 -> ring 2)

Driver: CCMC2 Z131C0.AK31 CM_MC_MECC_4 bit 149 Hi
Recver: BPR Z045B9.12 PDAT<2> bit 58 Lo

Pattern 0011 - dc_pat.011 FAILED 1 error

DC Connectivity Patterns FAILED: 6 errors

> Generating AC Patterns

AC Connectivity Patterns PASSED

Begin Full Test of Gate Arrays ...

> Generating patterns for MAUI at Z131K8

8.11.2 CST Failures - GA Locations.

- Gate array locations are given in a 3 part coordinate system.
 - Board side.
 - ◆ U is on the inside of the system board.
 - ◆ Z is on the back of the system board.
 - Distance from side of system board.
 - ◆ Counted left to right as viewed from the back side of the node.
 - ◆ The following table shows the numeric values of the gate array positions.

Table 14-1 Gate Array Locations

Gate Array	Location
AGENT	Z058
XBQ	Z073
XBS	Z102
MAUI	Z131
CCMC	Z131
SCI	Z163

- Distance from the bottom of the ASB/CSB.
 - ◆ The following table gives the alpha numeric designators for each slice.

Table 14-2 Slice Locations

Slice	Location
0	C0
1	F5
2	P4
3	T9

- CST Failure Ex.

Wire MCMB0MECC4 (ring 2 -> ring 2)
Driver: CCMC2 Z131C0.AK31
Recver: BPR Z045B9.12

CM_MC_MECC_4 bit 149 Hi
PDAT<2> bit 58 Lo

- The source is the CCMC at Z131C0.
 - ◆ Slice 0.
- The destination is the MTV on slice 0.
 - ◆ Location of the destination is given as a BPR (Bus Pain Reliever), this is part of the MTV and is also on slice 0.
 - ◆ Coordinate Z045B9.12 is a part (BPR) on the MTV, not on the system board!
- The ring (ring 2 in the above example) also determines the slice on which the failure was detected.

8.12 do_reset

do_reset [level [node ...]] - SPP MU Reset / Initialization Routine

- do_reset sends a message to one or more MUs, requesting that they perform one of three levels of "reset".
- All options are numbers.
 - The first number specified is the level of reset to do (see below).
 - If no number is specified, a "reset" level of 3 is assumed.
 - Any additional numbers specified are taken to be node numbers, and must be between 0 and 15 (inclusive).
 - If no node numbers are specified, the default is to send the message to all nodes in contact with the test station.
 - If a node number is to be specified, the "reset" level must be specified.
- The following reset levels are available:
 - level 1 - Hard reset of MU, MU Selftest, MU Init, Node Init
 - ◆ node init: Primary Loader and MU firmware are loaded
 - ◆ *power reset, pull*
 - level 2 - MU Init, Node Init
 - ◆ *power reset, pull*

- level 3 - Node Init (both the MU Node Init and the Primary Loader, as specified in the Config Parameters)
- Use of this utility will halt any activity taking place on the nodes being reset.

8.12.1 Sysreset Failure Message

The following error message from `do_reset` indicates that the checksum for the primary loader has gone bad.

```
elmo:/users/sppuser$ do_reset
do_reset: attempting a level 3 reset of node 0
do_reset: waiting 2 seconds to be sure reset is complete.
do_reset: checking for completion on node 0
do_reset: STATUSMU value of 0x00000694 indicates an
error condition, node 0
do_reset: TRACEMU: 0x00001200 INFO_MU: 0x00000001
errno is 0x694 (1684)
+++> 166
<Tue Apr 19 08:19:53 1994> info:0x456a0403
do_reset:0.0.37.200:../do_reset.c:290
```

- The error report seen above will also be stored in the `event_log`.

Note: See Appendix •G on page 266, the STATUSMU value of 0x00000694 indicates the type of failure.

8.13 Ecc_help - ECC Error Decode Utility

`ecc_help` - is a program to decode ecc error information.

- Depending on the type of CCMC installed in the SPP, the `-d` (for dark) may need to be used.
 - `ecc_help` - for CCMC s
 - `ecc_help -d` - for CCMC2s
- `ecc_help` prompts the user for three values that can be found in the `event_log (/spp/data/event_log)`.
 - the CAUSE csr
 - the ECC_ERR csr
 - and the ECC_ADDR csr.
- Once entered the program prints information that is contained in the various fields and identifies the failing bit and simm.
 - If the syndrome of the failure indicates that there was a greater than single bit failure - it will identify the 2 simms that are potentially in error.

Note: In some cases `ecc_help` may call out the wrong simm. Usually the simm at fault is on the other half of the long word read from memory. For example, if A0H is called and is not at fault, then the simm at A0L would be the next suspect.

- Example: /spp/data/event_log entry

```
+++> 344
<Thu Dec 8 16:14:38 1994> warning:0x67600400
<MU FW>:2.0.2.0:herr.c:842
node: 0x0 (0)
Soft Error received from CCMC 1
CAUSE: 0x02000028
CMC type is dark
0x02000000 - SCI incoming parity err
0x00000020 - single-bit ECC
0x00000008 - multiple ECC errors have occurred
ECC_ERR: 0x00018800, ECC_ADR: 0x00000040, CONFIG: 0x00020004
****
```

/spp/bin/ecc_help output

```
elmo_d:/users/sppuser$ ecc_help -d
Enter the ERR_CAUSE_0 CSR value : 0x02000028

Enter the MEM_ECC_ERR CSR value : 0x00018800

Enter the MEM_ECC_ADR CSR value : 0x00000040

Enter the CONFIG CSR value : 0x00020004
```

ECC addr register = 00000040 ECC err register = 0x00018800

The following ECC errors were detected :-
Single bit ecc error,
Multiple single bit ecc errors,

```
ECC error. - Node 0 Address 0x00000040
ECC code read = 0x18
ECC code generated = 0x80
Syndrome indicates an error on bit 35
Simm in error is 0AH ( location = Z082D5 )
```

8.14 `event_logger` - SPP Test Station Event Logger

`event_logger` - receives messages from diagnostic utilities (through the event daemon) running on the Test Station and writes them to the event log, for later review or processing.

- Normally, `event_logger` logs all events it receives to the default event log.
 - However, both the events processed and the location of the log may be modified by command line options.
 - Some of the options check fields in the event's condition code in order to determine whether the event should be logged.
 - Will grow to 4 MBytes, then it is moved to a `.old` file.
 - See the man pages for details.

8.15 Event_log

The /spp/data/event_log is the output of the event_logger.

- The MU power up self test also logs error messages in the event_log.
- event_log example:

```
+++> 300
<Sat Sep 24 17:13:29 1994> info:0x476c8021
<MU FW>:1.4.0.0:pne.c:1270
node: 0x0 (0)
power supply channel 0x21 (-4.5 clock logic) is OK
shut_up: -4047 warn_up: -4274 nom: -4499 warn_dn: -4724 shut_dn: -4948
last eight readings: -4497 -4497 -4497 -4497 -4497 -4497 -4497 -4497
****
+++> 134
<Sat Sep 24 17:13:29 1994> info:0x47600411
<MU FW>:1.4.0.0:subtests.c:71
node: 0x0 (0)
subtest passed - 0xfa0 (4000)
****
+++> 2766
<Sat Sep 24 17:15:53 1994> info:0x456af000
log_event:1.4.0.0:../log_event.c:96
-----
Hard error occurred at : Sat Sep 24 17:15:47 CDT 1994
NODE = 0 First Hard Error = 0008 Hard Error = 0008 Hard Error Mask = 1fff
-----
pce_util: 1.4 (Fri Aug 5 09:30:24 1994)
```

... The Rest of the error message is omitted ...

- See "Hard Error Register - Bit Definitions" on page 226 for a bit map list of the First Hard Error register.

8.16 iod - I/O Diagnostic shell

- Run this script before you run iod to initialize memory for the IOD.
 - `/spp/scripts/mem_init_tc`
- Next, start the iod shell:
 - `/spp/bin/iod`
- The following commands are entered inside IOD at the "iod" prompt.
 - `iod> set_node 1`
 - `iod> do /spp/scripts/iod/exec/[test name]`
- Test names:
 - **lf** - simple SIOP tests. (68sec)
 - **dma** - Makes sure that each channel can do a dma transfer. (5min 67sec)
 - **dv** - DV regression tests converted to iod test scripts. These tests are of the pass/fail variety. They test several things, but unless you did some of the dv work on the io board, you can't extract useful debug data from them. (7min 32sec)
 - **map** - Checks out the peripheral map registers. Since there are 4k registers per unit, this test takes some time. (18min 48sec)
 - **safe_xa** - XA io tests with safe reads from disks & DAT tape. (35+ min)
 - **safe_cd** - CD io tests with safe reads from disks & DAT tape. (35+ min)
 - **fddi** - FDDI basic tests. (30sec)

8.16.1 Configuring the Disk Test

- The disk test, `safe_xa`, runs a fixed set of disk drives.
 - In most cases it must be modified to run on the desired drives.
 - Viewing the `safe_xa` script will enable the user to gain an idea of which files to edited.
 - ◆ The `safe_xa` and `safe_cd` scripts are located in the `/spp/scripts/iod/exec` directory.
 - `/spp/scripts/iod/p3/disk/safe/xa/u0_s0_test`
 - ◆ This file is called by `safe_xa`, it only executes commands on unit 0/sbus0 scsi ids 2,4 and a (diags 7.0.1).

8.17 io_tc

- io_tc is an I/O diagnostic ran by the TCHIPs, this make it an "at speed" test.

```

elmo_d:/spp/bin$ io_tc
Starting io_tc at Tue Oct 18 10:22:05 1994

Memory boards available      : 0 1 2 3
I/O available for Node 0    : 1
Tchips available for Node 0: 0 2 4 6
Slices available for Node 0: 0 1 2 3
      Subtest 110 - Node Download Verification Test          0:00:05
Entry address : 0x11818
Text address  : 0x10000
Text size     : 0x8004
Text pointer  : 0x5000
Data address  : 0x31000
Data size    : 0x2000
Data pointer  : 0xe000
BSS size     : 0x108e8
0:00:08  passed
Subtest 1000 - MU Unit 0 Talk Test          0:00:00  passed
Subtest 1010 - MU Unit 1 Talk Test          0:00:00  passed
Subtest 1020 - Tchip Unit 0 Talk Test       0:00:00  passed
      Subtest 1030 - Tchip Unit 1 Talk Test   0:00:01  passed
      Subtest 1040 - Tchip General Talk Test  0:00:00  passed
      Subtest 1050 - Multi Tchip Talk Test   0:00:00  passed
      Subtest 1060 - Cerebus - 1 Proc Test   0:00:00  passed
      Subtest 1070 - Cerebus - Multi-Proc Test 0:00:01  passed
      Subtest 1080 - Cerebus - Fifo check - 1 Proc 0:00:00  passed
      Subtest 1090 - Cerebus - Fifo check - Multi-Proc 0:00:00  passed
Subtest 2000 - MEM U0 CSR Test              0:00:00  passed
      Subtest 2010 - MEM U0 RAM Test         0:00:18  passed
      Subtest 2020 - MEM U0 MAP Test        0:00:01  passed
      Subtest 2030 - MEM U1 CSR Test        0:00:00  passed
      Subtest 2040 - MEM U1 RAM Test        0:00:18  passed
      Subtest 2050 - MEM U1 MAP Test        0:00:01  passed
Subtest 3010 - MATS Test Pattern 2         0:00:03  passed
      Subtest 3020 - MATS Test Pattern 3    0:00:02  passed
      Subtest 3030 - MATS Test Pattern 4    0:00:03  passed
      Subtest 3040 - MATS Test Pattern 5    0:00:02  passed
      Subtest 3050 - MATS Test Pattern 6    0:00:03  passed
      Subtest 3060 - NTA Test Pattern 1     0:00:15  passed
      Subtest 3070 - NTA Test Pattern 2     0:00:16  passed
      Subtest 3080 - NTA Test Pattern 3     0:00:15  passed
      Subtest 3090 - NTA Test Pattern 4     0:00:17  passed
      Subtest 3100 - NTA Test Pattern 5     0:00:15  passed
Subtest 3110 - NTA Test Pattern 6         0:00:16  passed
      Subtest 4000 - Peripheral Map U0 Test  0:00:00  passed
      Subtest 4010 - Peripheral Map U1 Test  0:00:00  passed

```

Diagnostics & Utilities

Subtest 4020 - Peripheral Map U0 & U1 Test	0:00:00	passed
Subtest 5000 - DMA U0 Test	0:00:01	passed
Subtest 5010 - DMA U1 Test	0:00:00	passed
Subtest 5020 - DMA U0 & U1 Test	0:00:01	passed

Closing all T-chip channels..... please wait.....

INITWHAT value for Node 0 was 0x5b00df0f, restoring to 0x0000cf0f

+++> 188

<Tue Oct 18 10:24:54 1994> info:0x456b0405

io_tc:2.0.0.0:../main.c:70

terminating - testing passed

elapsed time: 02:49

Terminating normally at Tue Oct 18 10:24:54 1994

8.18 load_eprom

SPP Utility Board Eprom Write Utility

`load_eprom <node #> { -c | -f | -p | -s } <input_file>`

- `load_eprom` downloads a file to a section of the Utility Board EEPROM on a single node.
- `<node>` is the node to download to.
- The Utility Board must be running and in contact with the Test Station.
- Only one of the section specifiers may be specified. They have the following meaning:
 - `-c` utility board Core firmware (factory build option only)
 - `-f` utility board Firmware
 - `-p` Primary loader
 - `-s` Secondary loader (OS).. **OBP**

The file specified by `input_file` must be an image of the EEPROM section to be written and must be of the correct length.

- Used by the "restall" script.
- Counterpart is `unload_eprom`.
 - Used by the "saveall" script.

8.19 Reloading the MU firmware

- To reload the MU firmware from a file on the Test Station type:
 - `load_eprom 0 -f /spp/firmware/mu_firmware`
 - `sysreset 1 0`

8.20 Reloading the Primary Loader

- Reload the Primary Loader from a file from the Test Station.
 - `load_eprom 0 -p /spp/firmware/pl`

8.21 Reloading the Secondary Loader

- Reload the Secondary Loader from a file from the Test Station.
 - `load_eprom 0 -s /spp/firmware/obp`

8.22 New Versions of the load Commands

- `load_mufw 0` - loads MU Firmware to node 0
- `load_pl 0` - loads the Primary Loader to node 0
- `load_obp 0` - loads the OBP to node 0

Note: These script will load default factory versions of firmware, thus all parameter modifications will be lost.

8.23 Saveall - Saving NVRAM Contents.

`/spp/scripts/saveall [node #] [file_root_name]`

- generates 3 file
- EX: `/spp/scripts/saveall 0 node.0`

uploading secondary loader

```
+++> 237
<Thu Jan 26 09:58:25 1995> info:0x456a0000
/spp/bin/unload_eprom:2.0.0.0:../unload_eprom.c:180
going to upload 0x7f000 (520192) bytes from MU to node.0.nvm.obp
starting at address 0xf0880000 (-259522560) on node 0x0 (0)
****
```

0xf08fee00

```
+++> 160
<Thu Jan 26 09:58:34 1995> info:0x456a0401
/spp/bin/unload_eprom:2.0.0.0:../unload_eprom.c:295
terminating normally
file uploaded successfully
****
```

uploading primary loader

```
+++> 236
<Thu Jan 26 09:58:34 1995> info:0x456a0000
/spp/bin/unload_eprom:2.0.0.0:../unload_eprom.c:180
going to upload 0x40000 (262144) bytes from MU to node.0.nvm.pl
starting at address 0xf0840000 (-259784704) on node 0x0 (0)
****
```

0xf087fe00

```
+++> 160
<Thu Jan 26 09:58:38 1995> info:0x456a0401
/spp/bin/unload_eprom:2.0.0.0:../unload_eprom.c:295
terminating normally
file uploaded successfully
****
```

uploading MU firmware

```
+++> 239
<Thu Jan 26 09:58:38 1995> info:0x456a0000
/spp/bin/unload_eprom:2.0.0.0:../unload_eprom.c:180
going to upload 0x37000 (225280) bytes from MU to node.0.nvm.mu_fw
starting at address 0xf0808000 (-260014080) on node 0x0 (0)
****
```

0xf083ee00

```
<Thu Jan 26 09:58:43 1995> info:0x456a0401
/spp/bin/unload_eprom:2.0.0.0:../unload_eprom.c:295
terminating normally
file uploaded successfully
****
```

- Files generated in the current working directory:

```
-r--r--r-- 1 sppuser sppuser 520192 Jan 26 09:58 node.0.obp
-r--r--r-- 1 sppuser sppuser 262144 Jan 26 09:58 node.0.pl
-r--r--r-- 1 sppuser sppuser 225280 Jan 26 09:58 node.0.mu_fw
```

8.24 Restall - Restoring NVM contents.

- `/spp/scripts/restall [node #] [file_root_name]`
 - Completely restores the MU Non-volatile ram parameters.
 - Assumes you have the 3 files generated by `saveall`.
 - Reloads not only the parameters but the entire ram.

8.25 mp1_map - Address Decode Tool

Address to Hardware decode tool.

Memory configuration

DRAM Size(Mbit)	= 4
Virt -> Phys	= 0 1 2 3
Banks/board	= 2
Rows per simm	= 2
Boards	= 4
Interleave	= 4
Physical address	= 0x00000000
Node Num.	= 0x0
Board select	= mtv0
Bank select	= 0x0
Row select	= 0x0
Ram address	= 0x00000
Dram Row Addr	= 0x000
Dram Col Addr	= 0x000

- The following `sppring -s 10030` failure printout is an example where it become necessary to get the failing address and use `mp1_map` to determine which memory board is involved.

```

Subtest 10030 - CCMC Stress Test                                0:00:00
    Seeds used : 0x66c21c06 0x48135b98
0:00:30   failed
+++> 888
<Wed May 3 13:13:37 1995> error:0x856b0564
sppring:3.0.0.0:util.c:128
ERROR: Received unexpected trap from Node 1 CPU 7
    Expected trap : 0xffffffff   Actual trap : 0x1
HPMC:
Runway HPMC status0 0x0a608007 status1 0x44400144: path error
Processor SADD_LOG_TIDx register(s):
    CPU 7 TID 0   read_priv      0x04000280

APA Runway database:
Sys      CPU   Runway      Physical   Virt      Reply  Reply
TID CPU TID E Transaction Address     Indx F P Active RAM
  0  6   0  0 read_priv   0x04000780 0x00 0 0    0   0x5880
  1  7   0  1 read_priv   0x04000280 0x00 0 0    0   0x7029
  2  7   1  0 read_short  0xf0fceb40 0xf0 0 0    0   0x702a
  3  7   0  0 read_priv   0xffef0d60 0xff 0 0    0   0x702b

CPU 7 T 0xe851 E0 0xa771 E1 0x0139
T ERROR
E0 chip 0xa class 0x1 type 0x1 code 0x171
E1****

Exit - exceeded fail count limit.
INITWHAT value for Node 0 : 0x5b008fff
INITWHAT value for Node 1 : 0x5b008fff

Closing all T-chip channels..... please wait.....
+++> 194
<Wed May 3 13:16:06 1995> info:0x456b0406
sppring:3.0.0.0:../main.c:63
terminating - testing failed
elapsed time: 26:04
Terminating with 1 error at Wed May 3 13:16:06 1995

```

- Evaluating the failure message.

```

                                     Failing address
1   7   0  1 read_priv   0x04000280 0x00 0 0    0   0x7029

```

^

(The 1 indicates the failing request.)

• mp1_map shows

Memory configuration

```
DRAM Size(Mbit)      = 4
Virt -> Phys         = 0 1 2 3
Banks/board          = 2
Rows per simm        = 2
Boards               = 4
Interleave            = 4

Physical address     = 0x004000280

Node Num.            = 0x0
Board select         = mtv2 >>Memory board requested.
Bank select          = 0x0
Row select           = 0x0
Ram address          = 0x4000c
Dram Row Addr        = 0x100
Dram Col Addr        = 0x00c
Tag Dram Row Addr    = 0x100
Tag Dram Col Addr    = 0x00c
Tag Ras              = 0x0
```

8.26 pce_util - SPP Power, Clock & Environment Utility

NAME

pce_util - SPP Test Station Power, Clock, & Environment Utility

SYNOPSIS

```
pce_util [-c {s[source] <src> | mode] <mode> | on | off}]
[-d] [-D] [-l] [-L] [-n [<#>...]] [-p
{on|off|l[ower]|n[ominal]|u[pper]} [<#> or <list>] [-s
```

DESCRIPTION

pce_util processes the command line in order (left to right), sending messages to the appropriate MUs to accomplish the desired clock, power, or other environmental changes and/or report current status. With no options, pce_util defaults to displaying the current clock, power, and environmental state for each node in contact with the Test Station. When options are specified, this display of information usually occurs after processing all options

OPTIONS

- c manipulates the clocks
- d display the current state of all selected nodes
- D increment internal debug flag (expert use only)
- l when displaying state, include the set-point levels if no error

- L** when displaying state, include the set-points even with error
- n [<#> ...]** specify the nodes to work with. the default on start-up and if no numbers are specified is all nodes in contact with the Test Station.
- p <command> [voltages]** manipulate power on the current nodes. Each command is followed by a number specifying the voltages to be manipulated, or zero or more text voltage terms. If any terms are present, only the specified voltages will be manipulated, otherwise all voltages will be manipulated. If a memory or IO board voltage is specified to be turned on the boards is not installed, the voltage will not be turned on. Valid terms are: VEE_CLK, VHE_HP, VTT, VTT_SCI, VTT_GA, VDD_GA, VHC_HP, VDL, VCC_MB0, VCC_MB1, VCC_MB2, VCC_MB3, IO, VCC_HP

The '-p' must be followed by one of the following commands

- on** turn power on
- off[f]** turn power off
- l[ower]** margin power to lower settings
- n[ominal]** margin power to nominal settings
- u[pper]** margin power to upper settings

-s suppress the display of state that normally occurs before termination.

OTHER

The power and temperature set-points are fixed in a database, and are not changeable through this utility.

RETURNS

The pce_util utility normally returns 0, even if some voltages or temperatures are out of tolerance. If a serious error occurs, pce_util will return a non-zero value.

8.26.1 pce_util output

```
scri_d% pce_util -p off
pce_util: 1.3 (Fri Jun 17 09:58:08 1994)
```

```
node: 0x0 (0)
```

```
status description state current
```

```
-----
```

```
enabled clocks normal free running
```

```
off -4.5 VEE_CLK unknown 0.048
off -3.1 VHE_HP unknown 0.002
off -2.0 VTT unknown 0.000
off -2.0 VTT_SCI unknown 0.001
off +1.2 VDD_SCI unknown 0.001
off +1.2 VDD_GA unknown 0.001
off +2.0 VHC_HP unknown 0.001
off +3.3 VDL unknown 0.097
off +5.0 VCC_MB0 unknown 0.002
off +5.0 VCC_MB1 unknown 0.002
off +5.0 VCC_MB2 unknown 0.002
off +5.0 VCC_MB3 unknown 0.002
off +5.0 IO unknown 0.002
off +5.0 VCC_HP unknown 0.061
```

```
ok memory 0 temp 77F
ok memory 1 temp 74F
ok memory 2 temp 74F
ok memory 3 temp 73F
ok power 0 temp 73F
ok power 1 temp 75F
```

```
fan 1 (top) ok
fan 2 (middle) ok
fan 3 (bottom) ok
```

```
scri_d% pce_util -p on
pce_util: 1.3 (Fri Jun 17 09:58:08 1994)
```

```
node: 0x0 (0)
```

```
status description state current
```

```
-----
```

```
disabled clocks normal run w/jtag
```

```
on -4.5 VEE_CLK unknown -4.497
on -3.1 VHE_HP unknown -3.105
on -2.0 VTT unknown -2.001
off -2.0 VTT_SCI unknown 0.001
off +1.2 VDD_SCI unknown 0.001
on +1.2 VDD_GA unknown 1.199
on +2.0 VHC_HP unknown 1.997
on +3.3 VDL unknown 3.303
on +5.0 VCC_MB0 unknown 5.000
on +5.0 VCC_MB1 unknown 5.004
on +5.0 VCC_MB2 unknown 5.007
on +5.0 VCC_MB3 unknown 5.007
off +5.0 IO unknown 0.002
on +5.0 VCC_HP unknown 5.000
```

```
ok memory 0 temp 76F
ok memory 1 temp 77F
ok memory 2 temp 76F
ok memory 3 temp 75F
ok power 0 temp 75F
ok power 1 temp 77F
```

```
fan 1 (top) ok
fan 2 (middle) ok
fan 3 (bottom) ok
```

8.27 sn_cnsl

This utility allows you to spy on or gain control of the OBP console window.

- **sn_cnsl -S 1** - to snap-shot the console window on node 0
- **sn_cnsl -F 1** - to gain complete control of the console window
- **^e c .** - to exit either session
- **^e c f** - to regain control on the test station

8.28 SPP Diagnostic Shell

The `sppdsh` is a ksh based diagnostic shell. It must be invoked before using the following commands.

- **Use this shell when trouble shooting.**
- KSH with added features.
- `mget [address]` get the contents of a memory location
- `mput [address] [value]` put value information into memory location

Sppdsh provides an entire script language complete with flow control and logical operators. See the man page for more details.

To change nodes inside the sppdsh:

```
sppdsh$ node 1
```

8.29 Spring CxRing and Distributed Memory Test

8.29.1 Class 1 Access Verification subtests

- Class 1 subtests verify that the test station can access memory, CCMC, CxRing controller, and CxRing through MU. MAUI, XBAR, CCMC, and BPR. It also verifies that the network cache and error reporting circuits of CCMC are functioning correctly.

8.29.2 Class 2 Distributed Memory (Tag) Subtests

- Class 2 subtests check the integrity of the distributed memory system. Subtests in class 2 are run on all the specified nodes simultaneously by stuffing corresponding tag code from the primary loader PDC space to T-Chip cache.

8.29.3 Class 3 Distributed Memory (Data) Subtests.

- Class 3 subtests check the integrity of the distributed memory system. Subtests in class 3 are run on all the specific nodes simultaneously by stuffing corresponding memory code from the primary loader PDC to T-Chip cache (except memory cycles test).

8.29.4 Class 4 Non-Coherent Subtests

- Class 4 subtests check the non-coherent read/write functions of the system.

8.29.5 Class 5 Download Verification and EIR Check Subtests

- Class 5 subtests check the download and CPA EIR interrupt generation function of the system. It also tests the MAUI TOC and thread count circuit.

8.29.6 Class 6 Intranode Coherency Subtests

- Class 6 subtests check the intranode coherency function of the system.

8.29.7 Class 7 CxRing Internode Coherency Subtests.

- Class 7 subtests check the internode coherency function of the system.

8.29.8 Class 8 Message Subtest

- Class 8 subtest checks the messaging hardware of the system.

8.29.9 Class 9 SCI Stress Subtests

- Class 9 subtests perform sequences of load/store/flush to an internode address to stress the SCI node chips.

8.29.10 Class 10 Coherency Stress Subtests

- Class 10 subtests perform a random load/store/flush from each processor.

8.29.11 Class 11 SIMM Quick Sanity Subtests

- Class 11 is used by manufacturing to do a quick sanity check on tag and data address lines.
 - It does not require any tchip to be present.

8.29.12 Class 12 Tag SIMM quick check.

- Class 12 s used by manufacturing to do a quick check on the first 1KBytes of tag simms.
 - It does not require any tchip to be present.

8.29.13 Class 13 Data SIMM quick check.

- Class 13 s used by manufacturing to do a quick check on the first 1KBytes of data simms.
 - It does not require any tchip to be present.

*Note: Check the latest revision of the SPP ring manual (#700-00331300-000)for the current test class descriptions.

8.30 MU power on Diagnostic

- Eliminated most of the multiply and divide instructions. These instructions tend to cause the 68040 to not want control of its bus. This occasionally hangs the board because of a bug in the arbitration logic.
- The values in the MSIZE0 and MSIZE1 config parameters are stored in the CMC_NODE_MSIZ0 and CMC_NODE_MSIZ1 CSRs when the CMC arrays are initialized.
- Here's the current default set of diagnostic subtests.

Bus error test:

1000	source address data pattern subtest	
	enabled	
1010	force subtest	enabled
1020	read timeout subtest	enabled
1021	write timeout subtest	enabled
1030	source address functional subtest	enabled

Access register test:

1100	data pattern subtest	enabled
------	----------------------	---------

RAM test:

1200	quadlet data pattern subtest	enabled
1201	doublet 0 data pattern subtest	enabled
1202	doublet 1 data pattern subtest	enabled
1203	byte 0 data pattern subtest	enabled
1204	byte 1 data pattern subtest	enabled
1205	byte 2 data pattern subtest	enabled
1206	byte 3 data pattern subtest	enabled
1210	quadlet address subtest	enabled
1211	doublet address subtest	enabled
1212	byte address subtest	enabled
1220	byte selection subtest	enabled
1230	parity error subtest	enabled

Diagnostics & Utilities

Interrupt test:

2000	interrupt acknowledge subtest	enabled
2010	primary int data pattern subtest	enabled
2011	hard error int data pattern subtest	enabled
2012	secondary int data pattern subtest	enabled
2020	interrupt operation subtest	enabled
2021	interrupt combination subtest	enabled
2030	source subtest	enabled

DaRT interface test:

2100	data pattern subtest	enabled
2110	register address subtest	enabled
2120	access subtest	enabled
2130	interrupt subtest	enabled
2140	loopback subtest	enabled
2150	bus error and master-mode addr subtest	enabled

BBC test:

2200	data pattern subtest	enabled
2210	address subtest	enabled
2220	battery subtest	enabled
2230	interrupt subtest	enabled
2240	NVRAM uniqueness subtest	enabled

COP interface test:

2300	data pattern subtest	enabled
2310	loopback subtest	enabled

EEPROM access test:

3000	write-enable subtest	enabled
3010	protection subtest	enabled

Scan test:

3100	TBC data pattern subtest	enabled
3110	TBC address subtest	enabled
3120	access subtest	enabled
3130	reset subtest (broken subtest) DISABLED
3140	loopback subtest	enabled
3150	interrupt subtest	enabled
????	ring integrity subtest (requires gate arrays and power) DISABLED

COP interface test:

3200	access subtest	enabled
------	----------------	---------

Power and environment test:

3300	data pattern subtest	enabled
3310	ADC conversion control subtest	enabled
3320	ADC interrupt subtest	enabled

Refresh test:

3400	data pattern subtest	enabled
3410	functional subtest	enabled

CXRing configuration test:

3500	data pattern subtest	enabled
3510	loopback subtest	enabled

Clock control test:

3600	data pattern subtest	enabled
------	----------------------	---------

Burst-mode access test:

3700	burst copy subtest	enabled
3710	burst bus error subtest	enabled

LCD test:

3800	LCD read subtest	enabled
------	------------------	---------

MAUI interface test:

4000	data pattern subtest	enabled
4010	address subtest	enabled
4020	read byte selection subtest	enabled
4021	write byte selection subtest	enabled
4030	interrupt subtest	enabled
4031	hard error interrupt subtest	enabled
4040	access subtest	enabled
4050	read parity subtest	enabled
4051	write parity subtest	enabled

CXBar interface test:

5000	request hard error interrupt subtest	
 DISABLED (problem with IO interaction)	
5001	response hard error interrupt subtest	
 DISABLED (problem with IO interaction)	
5010	loopback write subtest	
 DISABLED (problem with IO interaction)	
5011	loopback read subtest	
 DISABLED (problem with IO interaction)	

8.31 node_ip_set

Command to change the node's ID's.

```
node_ip_set <serial_num> <node_id> <IP_addr>  
<UDP_base_port>
```

- <serial_num > is the node's serial number, as shown on its LCD.
- <node_id> is the node ID that you want to assign to the node.
- <IP_addr> is the node's IP address. Look in the /etc/hosts file to determine the correct IP address. For node 0, use the /etc/hosts file entry for mu_0000. For node 1, use mu_0001, etc.
- <UDP_base_port> should always be "675".

When the node has updated its node ID, it will reset itself as if you executed "do_reset 1".

8.32 verify_cables

verify_cables - a script to check the SCI cable connections

verify_cables writes to the PVT csr in Maui using SCI and then verifies the write via Dart the bus. Any data inconsistencies are highlighted so that the bad cable can be identified. A write is attempted from every node to every other node, in the order specified on the command line

Usage: `verify_cables <list of node nums> [-r <list of rings>] [-p <pattern>]`

Example:

This is an example of a failure between nodes 1 and 2 due to a bad cable.

```
elmo_d:/users/sppuser$ verify_cables 0 1 2
Default node is 0
Default ccmc is 0

*** RING 0:
Write from node 0 to 1.  Exp=5555aaaa, Act=5555aaaa, Diff=00000000
Write from node 0 to 2.  Exp=5555aaaa, Act=5555aaaa, Diff=00000000
Write from node 0 to 0.  Exp=5555aaaa, Act=5555aaaa, Diff=00000000

Write from node 1 to 2.  Exp=5555aaaa, Act=5555aaaa, Diff=00000000
Write from node 1 to 0.  Exp=5555aaaa, Act=5555aaaa, Diff=00000000
Write from node 1 to 1.  Exp=5555aaaa, Act=5555aaaa, Diff=00000000

Write from node 2 to 0.  Exp=5555aaaa, Act=5555aaaa, Diff=00000000
Write from node 2 to 1.  Exp=5555aaaa, Act=5555aaaa, Diff=00000000
Write from node 2 to 2.  Exp=5555aaaa, Act=5555aaaa, Diff=00000000

*** RING 1:
Write from node 0 to 1.  Exp=5555aaaa, Act=5555aaaa, Diff=00000000
Write from node 0 to 2.  Exp=5555aaaa, Act=5555aaaa, Diff=00000000
Write from node 0 to 0.  Exp=5555aaaa, Act=5555aaaa, Diff=00000000

Write from node 1 to 2.  Exp=5555aaaa, Act=5555aaaa, Diff=00000000
Write from node 1 to 0.  Exp=5555aaaa, Act=5555aaaa, Diff=00000000
Write from node 1 to 1.  Exp=5555aaaa, Act=5555aaaa, Diff=00000000
```

Diagnostics & Utilities

Write from node 2 to 0. Exp=5555aaaa, Act=5555aaaa, Diff=00000000
Write from node 2 to 1. Exp=5555aaaa, Act=5555aaaa, Diff=00000000
Write from node 2 to 2. Exp=5555aaaa, Act=5555aaaa, Diff=00000000

*** RING 2:

ERROR: writing address 0xffff85010
Write from node 0 to 1. Exp=5555aaaa, Act=5555aaaa, Diff=00000000
ERROR: writing address 0xffff89010
Write from node 0 to 2. Exp=5555aaaa, Act=deadbeef, Diff=8bf81445 FAIL ***
ERROR: writing address 0xffff81010
Write from node 0 to 0. Exp=5555aaaa, Act=deadbeef, Diff=8bf81445 FAIL ***

ERROR: writing address 0xffff89010
Write from node 1 to 2. Exp=5555aaaa, Act=deadbeef, Diff=8bf81445 FAIL ***
ERROR: writing address 0xffff81010
Write from node 1 to 0. Exp=5555aaaa, Act=deadbeef, Diff=8bf81445 FAIL ***
ERROR: writing address 0xffff85010
Write from node 1 to 1. Exp=5555aaaa, Act=deadbeef, Diff=8bf81445 FAIL ***

ERROR: writing address 0xffff81010
Write from node 2 to 0. Exp=5555aaaa, Act=deadbeef, Diff=8bf81445 FAIL ***
ERROR: writing address 0xffff85010
Write from node 2 to 1. Exp=5555aaaa, Act=deadbeef, Diff=8bf81445 FAIL ***
ERROR: writing address 0xffff89010
Write from node 2 to 2. Exp=5555aaaa, Act=deadbeef, Diff=8bf81445 FAIL ***

*** RING 3:

Write from node 0 to 1. Exp=5555aaaa, Act=5555aaaa, Diff=00000000
Write from node 0 to 2. Exp=5555aaaa, Act=5555aaaa, Diff=00000000
Write from node 0 to 0. Exp=5555aaaa, Act=5555aaaa, Diff=00000000

Write from node 1 to 2. Exp=5555aaaa, Act=5555aaaa, Diff=00000000
Write from node 1 to 0. Exp=5555aaaa, Act=5555aaaa, Diff=00000000
Write from node 1 to 1. Exp=5555aaaa, Act=5555aaaa, Diff=00000000

Write from node 2 to 0. Exp=5555aaaa, Act=5555aaaa, Diff=00000000
Write from node 2 to 1. Exp=5555aaaa, Act=5555aaaa, Diff=00000000
ERROR: writing address 0xffffc9010
Write from node 2 to 2. Exp=5555aaaa, Act=deadbeef, Diff=8bf81445 FAIL ***

Note: The "one liner" failure is most likely bogus whether seen before or after the ring with multiple failures:

8.33 verify_rings

verify_rings - verifies that the sci chips can be accessed from each node around the SCI ring.

Usage: verify_rings

Trouble Shooting

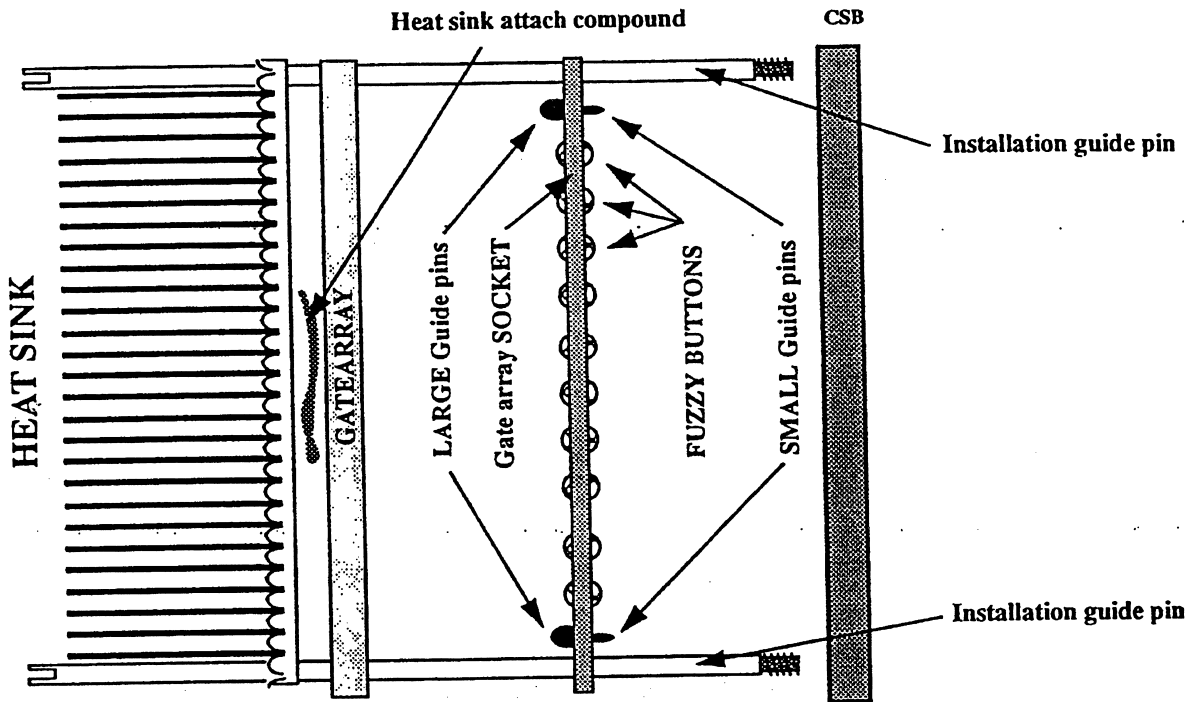
9

9.1 Gate Array Removal and Replacement

9.1.1 Removal and Replacement Parts

- Things you will need to correctly replace a gate array in the field are:
 - #1 Phillips head long shank screw driver
 - ◆ magnetic tip it *VERY* helpful.
 - The new gate array will be shipped with it's heat sink attached and a new socket.
 - Installation guide pins (Part number 315-000205-500).
 - Torque wrench capable of accurately gauging 7 inch/lbs.

Figure 9-1
Gate Array Removal and Replacement



9.1.2 Cleaning

- Whenever replacing a GA, it is suggested that the surface on the system board be cleaned.
 - The same tools used on the C3 and C4 series machines can be used on the CSB/ASB.
 - 91% or better alcohol.
 - A pure bristle paint brush trimmed to a length of about a half inch (not too stiff but firm enough to remove contamination).

- Cleaning the gate array should not be necessary.
 - Only if contamination can be visually detected.

9.1.3 Maintenance

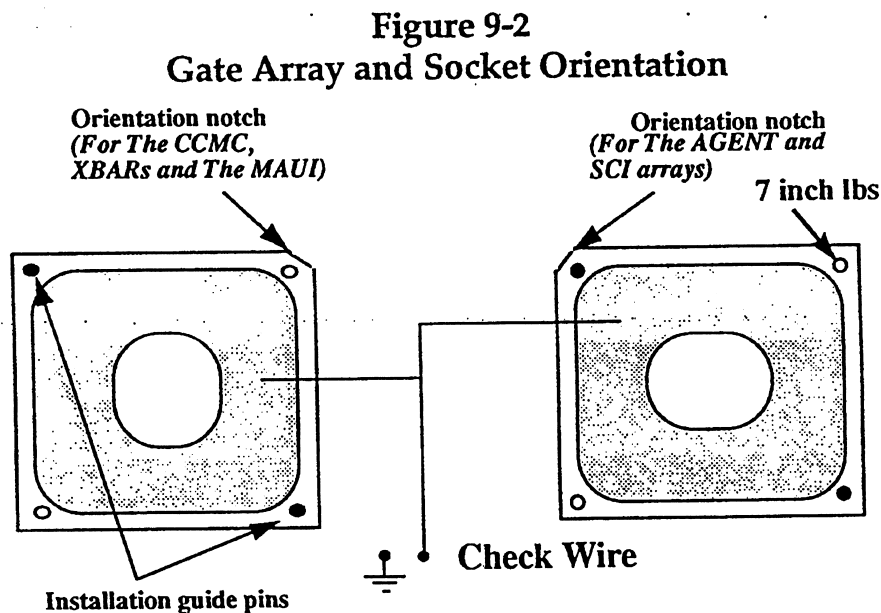
It has been seen that after an extended period of time, it is possible to detect a condition of insufficient torque on the gate arrays.

- Torque should be checked and reset to 7 inch/lbs at installation and every 90 days.
 - Checking torque should be done on any occasion where a connectivity problem is encountered.
 - Rotate from corner to corner when installing gate array screws, do not tighten screws in a circular order.

9.1.4 Orientation

Careful attention needs to be paid to the orientation of the gate array and the socket.

- Sever damage can result to the gate array, CSB and other devices on the CSB if the orientation is done incorrectly.



WARNING: On the Agent GA, it is possible to have it rotated 90 degrees out and still pass check wire.

Note: After verifying the gate array and socket to be properly oriented, also verify that the heat sink is properly positioned. The fins should be oriented horizontally in order to provide for proper cooling.

9.2 OS Hangs

9.2.1 When a System Hangs:

- Check for I/O hangs.
- Dump_all - checks for known problems.
- Turn off timeouts - eliminates train wrecks, allows a crashdump to be taken on a hung system.
- Crashdumps - crashsystem.

9.2.2 Identifying When the IO Board is Hung

The io board is divided into 2 units. If you think one of the units is hung and thus causing the system to appear hung, there is a simple procedure you can follow.

- Run the io_hang script.

```
sppdsh$ /spp/scripts/io_hang

Default node is 0
Default ccmc is 0
ERROR: reading address 0xffed0000
Landfill unit 1 is hung
Resetting the io board
Default node is 0
Default ccmc is 0
IO Channel state info for unit 0 chnl 3:
0xe0309003 0x00001004 0xff000800 0x70ff0000
0xff000800 0x0042ffff 0x00000000 0x80001afd
0x00000000 0x7ff7f4d4 0x7fc1ffdf 0xe7eddfef
0x80001afc 0x00000000 0x00000000 0x00000000
0x80005e1e 0x80005e1f 0x80005e20 0x80005e21
0x80005e22 0x80005e23 0x80005e24 0x80005e25
0x80005e26 0x80005e27 0x80005e28 0x80005e29
0x80005e2a 0x80005e2b 0x80005e2c 0x80005e2d
```

- Or manually read registers on the SPP to determine which unit on the SIOP is hung:
 - 1) Start up sppdsh.
 - 2) Make sure the xbar is working by reading an address in main memory. For example: "mget 0x1000"
 - If you get a time-out error, then your read request failed to work.
 - Since you can't communicate through the cxbar there is no way to talk to the io board.
 - 3) Try /spp/bin/do_reset to fix this problem.
 - 4) If the cxbar works, then you can check the io board by trying to read a register on one of its units.
 - For unit 0: "get 0xffec0000"
 - For unit 1: "get 0xffed0000"
 - If you get a value (any value) back, the io board is ok.
 - If you get the time-out error message, then the io board is hung.
 - Check the disk drives to see if any one drive has a light stuck on, thus indicating a possible error.

9.2.3 Hangs - Using dump_all

The dump_all utility can be used after a system hang to determine if any one of the three following problems might exist in the system.

- `/spp/scripts/cpa/dump_all node# | node_name`- run this script to dump the entire scan ring on the CPA.
 - `node#` - Are the actual node numbers e.g. 0 3 4.
 - `node_name` - Is the name assigned to a specific group of node.
- After running dump_all, the following three scripts are used to analyze it's output for certain problems:
 - `plr_hang` - source buffer deadlock
 - `check_for_ap bugs` - 1200 only
 - `cmp_chk` - cpc od spin error
- If any of the above scripts turn out to be positive, the TAC should be contacted for further actions.
 - Specific FMIs are in place to correct these type of failures.

9.2.4 Turning off Timouts

- The script "turn_off_timeouts" is located in /spp/scripts.
 - Use this script to keep the system from having a "train wreck" the next time it hangs.
 - With timeouts enabled, when the system goes idle for a period of time it will crash.
 - This causes multiple hard errors to occur, a train wreck, thus destroying the machine state.
 - When this happens it is necessary to turn off timeouts in order to get a crashdump.

9.2.5 Crashdumps

- The current method for taking a crashdump is to use the "crashsystem" program.
 - Example: elmo_d:/spp/bin\$ crashsystem -c [#]
 - Where "#" is either the node number or "all" for the entire system.
- After a crashdump partition has been dumped, it can be written out to a normal file using the "crashutil" program in SPP-UX.
- Crashsystem will not dump if you have more than 1 partition on a node marked as a crash partition.
 - Use diskutil to delete the "C" flags on all but one partition per node. See "Unset Partition Flags and Descriptions" on page 234

9.3 Hard Error Debug

9.3.1 Determining the Source of a Hard Error

Hard errors are detected throughout the system by the various gate arrays and IO boards.

- Once an event happens, the subsystem detecting the error reports to the MU that a hard error has occurred.
 - Registers on the MU board are then set indicating which functional unit detected the error.
 - The test station is notified that a hard error is present.
 - The front panel display is set to indicate that a hard error has taken place.
- The error is further defined by the `hard_logger` which is automatically executed by the `event_logger`.
 - Both the "first hard error" and "hard error interrupt mask" registers are examined and and'ed together to determine which interrogators to run.
 - ◆ Interrogators scan additional error register information from the parts involved and define a list of extractors to be executed.
 - ◆ Extractors literally extract all the pertinent information for a particular failure.
 - Results are sent to the `event_log` file.
- This only happens if the `MUERRENABLE` bit is set in the `INITWHAT` config parameter.

9.3.2 Manually Extracting Error Information

Some systems that appear to be hung may actually have a hard error set.

- To manually check for any hard errors, use "mget" to view the register 0xf0c0405c.
 - This is the First Hard Error register.
 - Mget must be executed inside sppedsh.
 - If any bits are set in this register, then a hard error does exist.
 - Use the following table to decode this register:

Table 9-1 Hard Error Register - Bit Definitions

Bit	Type of Error
15-13	Not used - reads as 0's
12	maui_harderr
11	io_harderror[1] - Unit 1
10	io_harderror[0] - Unit 0
9	xbar_response_harderr
8	xbar_request_harderr
7	agent[3] harderr
6	agent[2] harderr
5	agent[1] harderr
4	agent[0] harderr
3	ccmc[3] harderr
2	ccmc[2] harderr
1	ccmc[1] harderr
0	ccmc[0] harderr

- If there is a hard error, check the file `"/spp/data/event_log"` to see what the `hard_logger` has found.
 - In some cases it may be necessary to run the `hard_logger` manually.
 - Use the `-f` option to force it run. In some cases the hard error interrupt mask may have been cleared.
 - At the test station prompt type: `hard_logger -f`.

9.3.3 Agent Hard Error

The following agent hard error is an example of what might be found in the event_log after a system crash.

- Along with hard error information, there will also be environmental as well as hardware configuration data displayed within the event_log.

```
<Fri Jan 19 08:54:40 1996> error:0x856e0f02
log_event:7.0.1.0:../log_event.c:96
```

```
-----
Hard error occured at : Fri Jan 19 08:54:29 CST 1996
NODE = 0 First Hard Error = 0020 Hard Error = 0022 Hard Error Mask = 1fff
```

```
First Hard Error Decoded = AG1
-----
```

```
pce_util: 7.0.1 (Thu Sep 14 10:53:36 1995)
```

```
node: 0x0 (0)
```

status	description	state	current
-----	-----	-----	-----
disabled	clocks	normal	free running
on	-4.5 VEE_CLK	unknown	-4.497
on	-3.1 VHE_HP	unknown	-3.107
on	-2.0 VTT	unknown	-1.999
off	-2.0 VTT_SCI	unknown	0.001
off	+1.2 VDD_SCI	unknown	0.001
on	+1.2 VDD_GA	unknown	1.199
on	+2.0 VHC_HP	unknown	2.008
on	+3.3 VDL	unknown	3.298
on	+5.0 VCC_MB0	unknown	5.017
on	+5.0 VCC_MB1	unknown	5.014
on	+5.0 VCC_MB2	unknown	5.014
on	+5.0 VCC_MB3	unknown	5.004
on	+5.0 IO	unknown	5.004
on	+5.0 VCC_HP	unknown	5.012
ok	memory 0 temp		84F
ok	memory 1 temp		78F
ok	memory 2 temp		76F
ok	memory 3 temp		77F
ok	power 0 temp		80F
ok	power 1 temp		77F
	fan 1 (top)		ok
	fan 2 (middle)		ok

fan 3 (bottom) ok

Welcome to the Convex Configuration Manager Utility

ccmu: 7.0.3 (Tue Sep 26 19:25:36 1995)

ccmu: defaulting to the following nodes: 0

node	cop	id	type	pn	sn	ar	wr	sc	ac	dc	xl	
0	A0	0	TCHP	000001	1097465							
0	B0	1	TCHP	000001	1097347							
0	A1	2	TCHP	000001	1097467							
0	B1	3	TCHP	000001	1097350							
0	A2	4	TCHP	000001	1097348							
0	B2	5	TCHP	000001	1097349							
0	A3	6	TCHP	000001	1097468							
0	B3	7	TCHP	000001	1097466							
0	MB0	8	MTV	001374	1113017	b	b	1	1	1	0	
0	MB1	9	MTV	001374	1999253	a	a	1	1	1	0	
0	MB2	10	MTV	001374	1999263	a	a	1	1	1	0	
0	MB3	11	MTV	001374	1999254	a	a	1	1	1	0	
0	IO	12	LM	002376	1999963	a	c	1	1	1	0	
0	CSB	13	CSB4	001396	1999770	d	b	1	1	1	0	csn: 65551
0	MU	14	MU	001375	1999026	b	a	1	1	1	0	node #: 0

+++> 662

<Fri Jan 19 08:54:44 1996> error:0x856e0111

log_event:7.0.1.0:../log_event.c:96

 NODE 0 CPA 1 detected a Data queue read parity error

Values

Scan Fields

00 20 31 19

paq.pdq.r_d

1 1* 0 0

paq.pdq.r_p

0x00040004

csri.csr_local.csr1_regs.r_hard_err_lsw

Source of transaction is Proc_1

 * indicates parity error

Proceedurs and Trouble Shooting Aids

```
+++> 1597
<Fri Jan 19 08:54:46 1996> error:0x856e0136
log_event:7.0.1.0:../log_event.c:96
```

```
-----
NODE 0 CPA 1 PROC 2 HPMCH - Error Message Received
-----
```

```
Hourini Database Processor 0
-----
```

```
E1
Major   Minor   Source   TID
  6       0       2       0
-----
```

```
E0
ChipID   Class   RQ   Error Code
  a       1     1     86
-----
```

```
-----
V E R O M J M N P 0   P 1   P 2 L V I T M I P I C T O R S B
0 1 1 8   0 0   0003 4bc3 4   b 4b 0 0   0 0   0 0
1 1 1 f   1f f   0003 ffff a   a aa 0 0   0 0   0 0
2 1 0 4   0 0   0003 b0d4 b   0 b0 0 0   0 0   0 0
3 0 0 8   0 3   0002 f602 f   6 f6 0 0   1 0   0 0
-----
```

```
-----
Selected Byte Write Data: 002000fa
-----
```

(more information on "Hourini Database" has been omitted, for beivity)

- The "first hard error" register indicates that the Agent in slice 1, on node 0, has detected a hard error.
 - After the CCMU displays it's information, the hard error is further defined as a data que read parity error.
 - From Table 5-1, "Agent Internal Errors," on page 82, it is determined that this is an internal error.

9.3.4 MAUI Hard Error

Figure 9-3
MAUI Hard Error

```

++> 3188
<Fri Feb 16 11:11:53 1996> error:0x856e0f02
log_event:7.0.1.0:../log_event.c:96
-----
Hard error occurred at : Fri Feb 16 11:11:38 CST 1996
NODE = 0 First Hard Error = 1000 Hard Error = 1000 Hard Error Mask = 1fff
First Hard Error Decoded = MAUI
-----
pce_util: 7.0.1 (Thu Sep 14 10:53:36 1995)

```

node: 0x0 (0)

status	description	state	current
disabled	clocks	normal	free running
on	-4.5 VEE_CLK	unknown	-4.497
on	-3.1 VHE_HP	unknown	-3.107
on	-2.0 VTT	unknown	-1.999

(power parms and cops are omitted for brevity)

```

<Fri Feb 16 11:11:59 1996> error:0x856e0e02
log_event:7.0.1.0:../log_event.c:96
-----

```

NODE 0 MAUI detected a MURI read parity error
err_glob.error_sticky_csr[12] = 1

Scan Fields

La1 Start Xfer = 0	clk_mi.mso_ctl.la1_start_xfer
La2 Start Xfer = 0	clk_mi.mso_ctl.la2_start_xfer
La3 Start Xfer = 0	clk_mi.mso_ctl.la3_start_xfer
La1 First Load = 0	clk_mi.mso_ctl.la1_first_load
La2 First Load = 0	clk_mi.mso_ctl.la2_first_load
La3 First Load = 0	clk_mi.mso_
La1 Rd Add Hld = 0	clk_mi.mso_ctl.la1_rd_addr_hold
La2 Rd Add Hld = 0	clk_mi.mso_ctl.la2_rd_addr_hold
La3 Rd Add Hld = 0	clk_mi.mso_ctl.la3_rd_addr_hold
Rd Addr = 0x00	clk_mi.mso_ctl.rd_addr
Read Data [0:31] = 47 00 00 00	bs_stf.muri.ram0.rdata
Read Par [0:3] = 1 1 1 1	bs_stf.muri.ram0.rpty
Stg16 data [0:31] = 4700 0000	clk_mi.mso_path.stg16_dat

Proceedurs and Trouble Shooting Aids

La8 data [0:15]	= 0000	clk_mi.mso_path.la8_dat
Mi Out Tag	= 0x03	clk_mi.mso_ctl.mi_out_tag
Select Ram	= 1	clk_mi.mso_ctl.sel_ram
Select la8	= 0	clk_mi.mso_ctl.r_sel_la8

- The first hard Error indicates that this is a MAUI hard error in node 0.
 - The following error log information indicates which part of the MAUI detected the error.
 - NODE 0 MAUI detected a MURI read parity error.
- From the list of internal and external MAUI errors, (See "MAUI Hard Errors" on page 81) it can be determined that this is an internal error to the MAUI.
 - CST should be ran anyway.
 - Change the MAUI gate array.

9.4 “bio_reap_status” Message

- The “bio_reap_status” error messages which appear in the system console window are caused by I/O hardware problems.
 - They may cause the system to crash or hang.
 - In most cases, the disk or scsi controller is at fault.
 - Any of the following devices can cause this type of failure:
 - ◆ scsi disk
 - ◆ scsi controller
 - ◆ ancot board
 - ◆ SIOP.
- The device number reported in the message corresponds to a disk partition or stripe.
 - To determine which disk partition or stripe is being reported:
 - ◆ Break the device number into the major and minor device numbers. The least significant 24 bits are the minor number. The remaining 8 bits are the major number
 - ◆ Then find the matching block device file in /dev/dsk.

- Example:

[73000001 001cb5f6 0:1] bio_reap_status: operation timed out, dev 0xb000009

- The device is 0xb000009
 - ◆ Minor number = 000009
 - ◆ Major number = b
- The corresponding file is:

```
brw-r----- 1 root  sys  11 0x000009 Apr 7 1995 stripe0
```

10.1 Booting SPP-UX

- OBP parameters affecting the boot drive and root partition can be viewed by using the “printenv” command at the OBP prompt.
 - boot-device
 - boot-directory
 - boot-file
 - boot-arg

10.1.1 Booting from Disk

- At the sppuser prompt on the Test Station:
 - do_reset
 - wait for the OBP to become active.
- In the OBP window (System Console).
 - [0:2] ok boot

10.1.2 Boot from the Dat Drive

- Install a bootable Dat Tape in the drive.
- In the OBP window type:
 - ok (2,0) boot rmt0

10.2 Disk Administration Under SPP-UX

10.2.1 Determining Drive Mapping

As seen during boot, it can be determined if a disk drive has already been mapped and where it's physical location is.

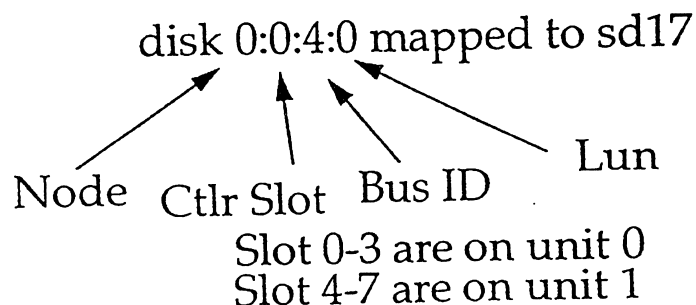
- A drive that is mapped will retain the same drive number no matter which physical hardware ID it is assigned.

- Mapping a drive, "labels" the drive with the desired drive number.
- Example of a disk drive that is already mapped: (as seen during boot)

```
[62000001 001e2500] scsi disk: disk 0:0:4:0 attached mapped to sd17
```

- Example of an unmapped drive:
[62000001 001e3288] scsi disk: disk 0:0:5:0 not mapped;

- The physical location of a disk drive can be determined from the above line printed during the boot processes or from diskutil while in OS.
 - The event_log will also have a record of all drives mapped during the boot process.
 - EX:



10.2.2 Mapping a Disk to a Logical Device

In SPP UX execute the program: `/etc/diskutil`

- Usage: `MAP Disk sd node:ctlr:tgt:lun` to device
- Diskutil: `MAP Disk sd 2:7:4:0` to `sd777`
 - This will map the disk drive in node 2's SIOP unit 1, sbus controller 3 at scsi ID 4 to `sd777`.

10.2.3 Unmapping a Disk

In SPP-UX execute the program: `/etc/diskutil`

- USAGE: `UNMAP Disk sd node:ctlr:tgt:lun` device
- Example: Diskutil: `UNMAP Disk sd 0:0:5:0`
 - Drives will not unmap if any partitions are marked active.
 - Check to see if a partition is active:
 - ◆ `DiskUtil: sel disk (sd###)`
 - ◆ `DiskUtil: show par`
 - If the partition is active it will have an "*" under the flags column.
 - A paging partition must first have it's flag removed (see 10.2.4), then reboot the OS to mark it inactive.
 - Unmounting mounted partitions will mark them inactive.

10.2.4 Unset Partition Flags and Descriptions

- **Diskutil: UNSet Par b Description**
 - Removes the partition description.
- **Diskutil: UNSet Par b Flag d**
 - Removes the default pager flag from partition b.
 - Substituting "d" at the end of the command line with "c" will remove the crashdump flag.

10.2.5 Set Partition Descriptions and Flags

- **Diskutil: SET Par a Description "describe me"**
 - Describe partition "a" with up to 30 characters.
 - Be sure to use the quotes.
 - Warning - exceeding 30 characters may destroy partition size information!
- **Diskutil: SET Par a Flag c**
 - Sets the crashdump flag on partition a.
 - Substituting "c" with "d" will mark it as a crashdump partition.

10.2.6 Displaying Disk Partitions

- The following illustrates how to display drive partitioning:

DiskUtil: select disk sd0

DiskUtil: show partitions

○ partitioning for the 4 GIG disk drive:

Logical disk name: sd0

partition table: (space available for file systems = 4194157)

part	offset	size	partition description	flags
a:	0K	819200K	Root and /usr filesystem	
b:	819200K	102400K	crashdump partition	C
c:	921600K	30720K	miniroot file system	
d:	952320K	1144832K	Default Pager Partition	D
e:	2097152K	2097005K	user space	

○ partitioning for the 2 GIG disk drive:

Logical disk name: sd0

partition table: (space available for file systems = 2098752)

part	offset	size	partition description	flags
a:	0K	819200K	root and /usr	
b:	819200K	102400K	ingres log	
c:	921600K	40960K		
d:	962560K	1024000K	paging partition	D

10.2.7 Creating a Partition:

- Diskutil:
 - DiskUtil: select disk sd3
 - DiskUtil: MAKE Partition a Size 1000M Offset 0
 - ◆ Will create an "a" partition of size 102400K, starting a beginning of the disk.
 - DiskUtil: MAKE Partition b Size 1000M after "partition #"
 - ◆ Will create a "b" partition after the end of the partition identified by "partition #".
- Either "offset" or "after" can be used to designate the beginning of the partition.

10.2.8 Deleting a Partition:

- Diskutil:
 - Diskutil: MAKE Partition a Size 0
 - ◆ Will delete partition a.

10.2.9 Striped Partitions

- First build individual partitions of the same size using `diskutil`.
- The actual stripe partition is also made using `diskutil`:
 - `diskutil: create stripe [name] [Blocksize<size>]`
(`<partition_name>` `<partition_name>` ...)
- `diskutil: destroy stripe [name]` - removes the stripe partition.
- Use `cnx_newfs` to make the file system if it is more the 2 Gbyte.
- `dumpfs` - Displays file system information.

10.2.10 NEWFS

After partitions have been defined on the disk drive, they must be "newfs'ed" before they can be mounted.

- These are a few examples of how to enable a partition for use, using the newfs command:
 - `newfs -v /dev/rdisk/sd3a scalios`
 - `newfs -v /dev/rdisk/sd3b scalios`
 - Scalios is a flag used to tell newfs to read the geometry off the drive instead of using disktab.
 - `cnx_newfs` - for file systems over 2 GIG Bytes

10.2.11 Mounted File Systems

- **mount -a** - Mounts all file systems found in */etc/checklist*.
- **umount [file system name]** - Unmounts a single file system.
- **bdf** - Displays file system information as percentages of available, total and used free space.
 - Mirrors what df does in Convex-OS.

10.3 SPP-OS FYI

10.3.1 Remote console commands

- Take a snap-shot of the OS console (spying).
 - `rlogin [hostname] -l sppuser` - Login to the test station as sppuser.
 - `/spp/bin/sn_cnsl -S [#]` (`#`= node number plus 1)
 - `^e c .` to exit
- Take control of the OS console.
 - Rlogin to the test station as sppuser.
 - `/spp/bin/sn_cnsl -F 1` - Gains control of system console.
 - `^e c .` - To exit.
 - `^e c f` - Regain control of console if needed.

su-cnsl -u list all virtual consoles

10.3.2 Passwords

- Standard OS root password "Acme".
- Standard Test Station root password "serialbus".
- sppuser password "spp user".

10.3.3 Show Processes

- `ps -edafl` (don't forget the "--")
 - Displays a long list of all processes running in UX.
 - Same on test station as in SPP-UX.

10.4 Important Files

- Boot files:
 - /os/mach
 - /os/server
 - /os/tuneables
- Network Files:
 - /etc/hosts
 - /etc/netlinkrc
 - /etc/src.sh
 - /etc/src.csh
 - /etc/set_parms
- File system:
 - /etc/checklist - lists file systems to be mounted at boot time or with "mount -a".
- SPP-UX version checks:
 - vers /os/mach
 - vers /os/server
 - sysinfo -vm (for the mach)
 - sysinfo -vs (for the server)
 - sysinfo -a (all available information)

Appendix • A Class Documentation list

- **Maintenance Manual**
- **SPPRing and Distributed Memory test manual**
Part Number 700-033130-000
- **Exemplar/CD Site Prep Guide**
Part Number 081-026230-001
- **Exemplar/XA Site Prep Guide**
Part Number 081-022830-001
- **Exemplar Architecture Manual**
Part Number 081-023430-001
- **Guide to Exemplar Documentation**
Part Number 710-029130-000
- **Wiring Diagram**
Doc number 550-000053-300
- **IO Power Backplane Drawing**
Doc number 410-002442-300B
- **Power Concentrator Card Drawing**
Doc number 410-001377-300A
- **Exemplar Open Boot Quick Reference**
Part number 710-029430-000

Appendix •B Tools

- 315-000205-500 INSTALLATION ROD, 4ea. - 2 sets.
- 4 would be needed to install scam cards-
- 7/16-9/16" socket - lowering feet on the towers.
- 1/8" socket - installing SCSI bus cables at the I/O chassis.
- Long, magnetized #1 Phillips screw driver - R&R on gate arrays.
- Set of #1 and #2 phillips and standard screw drivers - misc.
- 1/4 inch 2-16 in/lbs adjustable torque wrench - gate arrays.
- Allen wrench set - removing skins.
- 91% or better alcohol and soft bristle cleaning brush
- cleaning gate array connection surfaces.
- DVM - checking line voltage on install and check wire after gate array replacement.

Appendix • C FRU List

- **SPP1000_DEPOT_SPARES**

410-001396-200 BD ASSY, CSB4

- **ASICs**

182-000185-600 GATE ARRAY, SCI2

182-032730-600 GATE ARRAY, MAUI (SPP)

182-033930-601 GATE ARRAY, CPB (SPP)

182-000217-600 GATE ARRAY, CMCC2 (SPP)

182-039230-600 GATE ARRAY, CXBAR (SPP)

- **Test Station**

217-000012-200 WORKSTATION, HP 712 (CONVEX CONFIG)

217-000012-008 2GB DDS-1 EXTERNAL DAT DR, HP712

604-010011-200 CABLE ASSY, DIAG&REMOTE TEST DBL SH

- **Misc.**

410-001434-200 BD ASSY, NODE STATUS

410-002430-200 BD ASSY, MOD CONTROLLER

410-002431-200 BD ASSY, MOD LED SIDE

410-002432-200 BD ASSY, MOD LED TOP

500-000073-200 ASSY, LCD, SUPPORT

602-680001-201 CABLE ASSY, SCSI, 10 FT

601-200008-200 CABLE ASSY, MOD LED, TOP

601-200009-200 CABLE ASSY, MOD LED, SIDE

601-200010-200 CABLE ASSY, CSB TO LCD DISPLAY, LWR

601-200010-201 CABLE ASSY, CSB TO LCD DISPLAY, UPR

601-250008-200 CABLE ASSY, MU TO MOD CNTLR, LWR

601-250008-201 CABLE ASSY, MU TO MOD CNTLR, UPR

We will need to add additional cables as part numbers are identified.

- **SPP1000_AREA_SPARES**

550-000057-200 PROCESSOR MODULE, 100 MHZ W/1MB

410-002375-200 BD ASSY, MU2

410-001374-200 BD ASSY, MTV W/O SIMMS

410-002376-200 BD ASSY, I/O LANDMARK

- **Power**

410-001377-200 BD ASSY, POWER CONCENTRATOR

410-001378-200 BD ASSY, PWR BRICK, PWR0

410-001379-200 BD ASSY, PWR BRICK, PWR1

410-002380-200 BD ASSY, PWR BRICK, PWR2

410-001442-200 BD ASSY, CIOPC

• **Peripherals and Controllers**

550-000063-200 DAT TAPE ASSY, SCSI I/F, TSTD
550-000064-200 DISK DR, SE F/W SCSI 2GB, TSTD/FRMT
550-000065-200 ADAPTER, SBUS-SCSI2, DIFF F/W TSTD
550-000066-200 FDDI CNTRL, FIBER OPTIC, SBUS, TSTD
550-000423-200 CONV BRD, SE-DIFF F/W

• **Misc.**

500-000071-200 FAN ASSY, MPP NODE FAN
500-000074-200 ASSY, FAN, IO CHASSIS

We will need to add additional cables as part numbers are identified.

• **SPP1000_SITE_SPARES**

170-000001-001 RAM MODULE, 1MX36 60 NSEC SIMM
170-000002-001 RAM MODULE, 2MX36 60 NSEC SIMM
170-000003-001 RAM MODULE, 4MX36 60 NSEC SIMM
170-000004-001 RAM MODULE, 8MX36 60 NSEC SIMM

• **Power**

500-000055-200 ASSY, POWER SUPPLY CAMELOT 48Volt
200-001047-202 CONVERTER, 48VDC TO 2.0VDC 40A MAX
200-001049-200 CONVERTER, 48V IN 1.2V OUT DRIVER
200-001049-201 CONVERTER, 48V IN 1.2V OUT BOOSTER
200-001047-209 CONV, 48V/5V 200W SHT STUD DRIVER
200-001047-210 CONV, 48V/5V 200W SHT STUD BOOSTER
200-001047-211 CONV, 48V IN 12V OUT 200W BOOST
200-001047-212 CONV, 48V IN 12V OUT 200W DRV

• **Fuses**

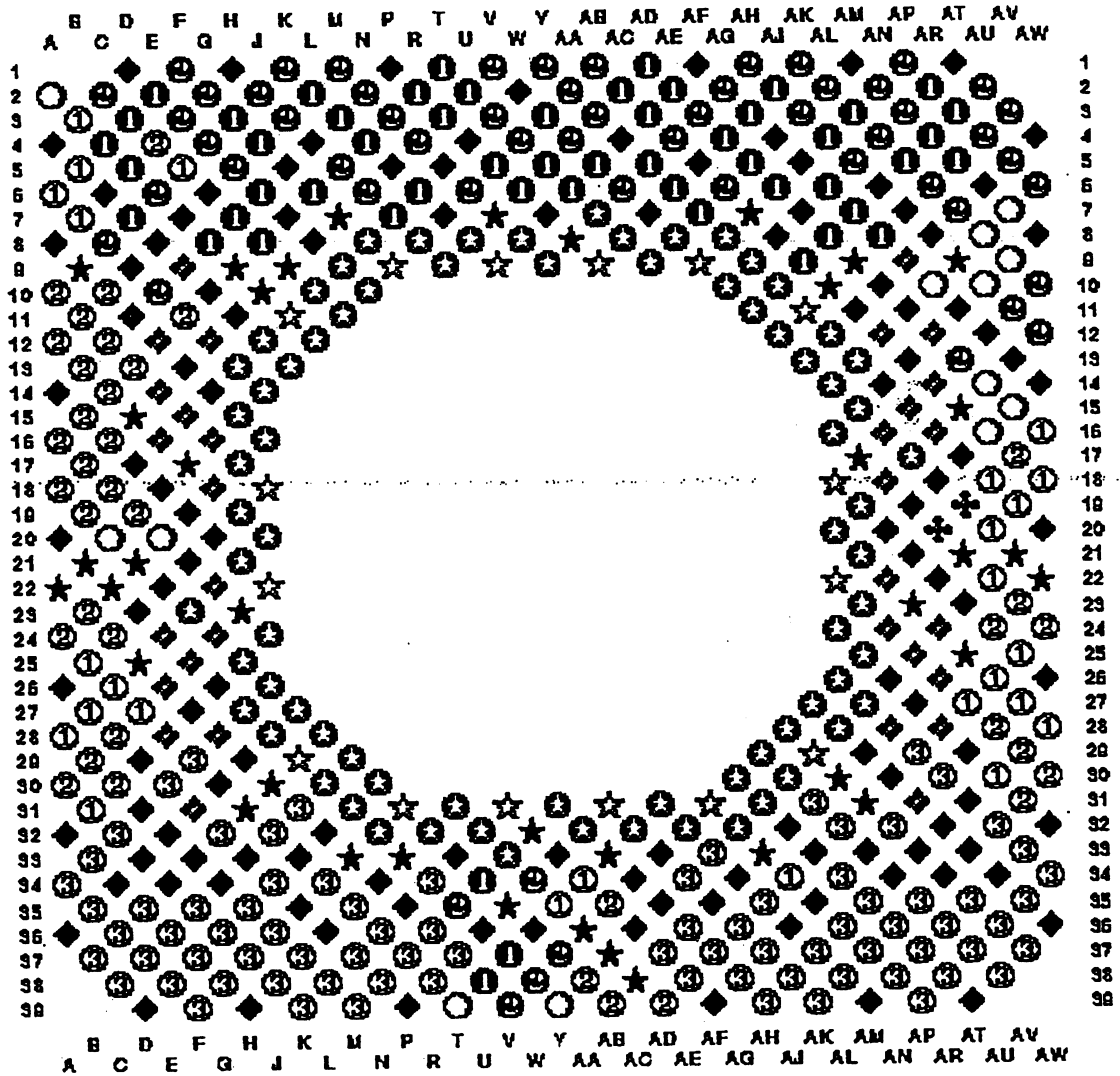
253-000124-001 FUSE, 3A 250V, PC-TRON
253-000124-003 FUSE, 5A 250V, PC-TRON
253-000128-001 FUSE, 10A/250V.25X1.25 FAST CER

We will need to add additional cables as part numbers are identified.

Appendix •D Gate Array pin-out

1) Agent

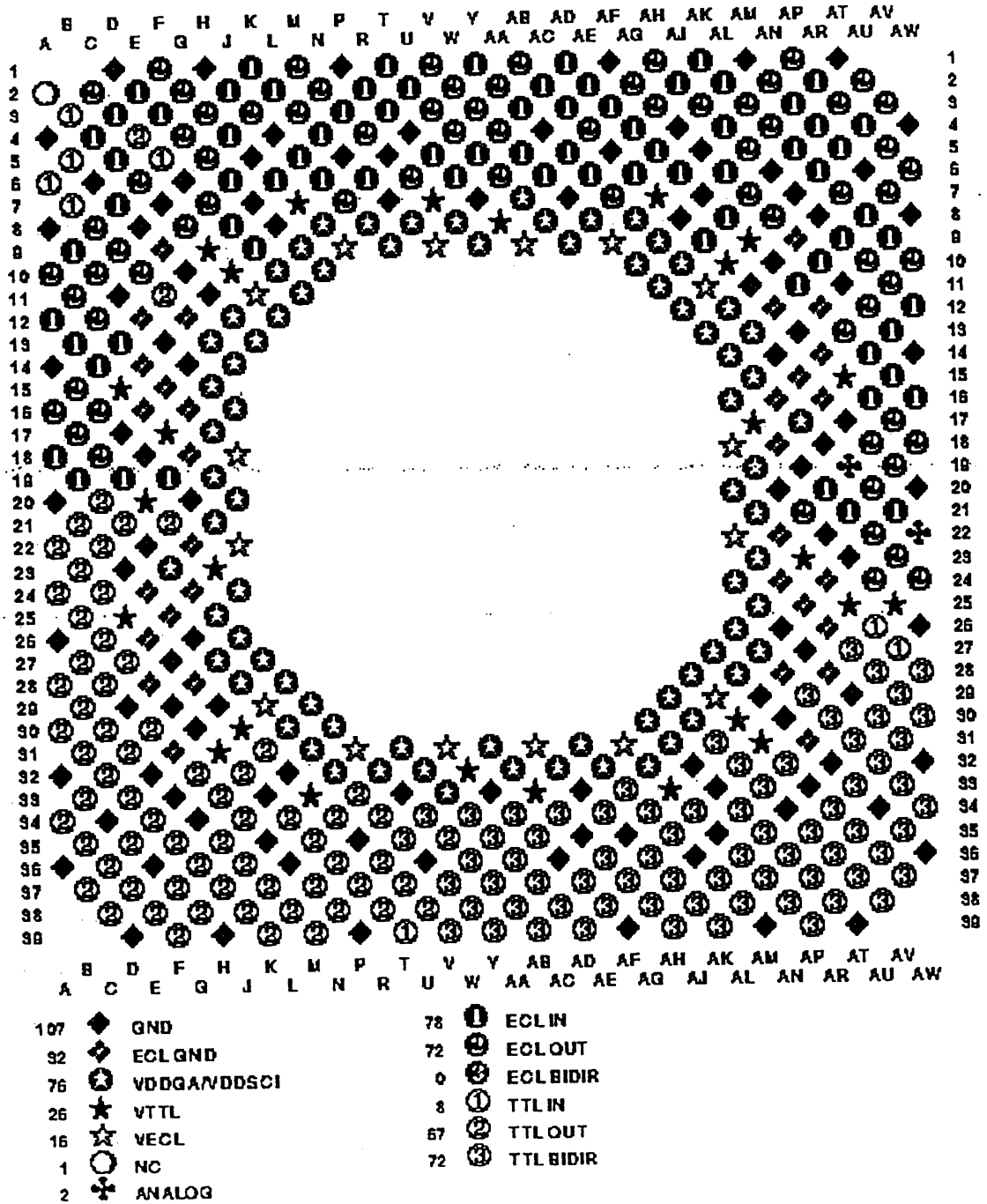
LGA557 PIN MAP FOR AGENT [probe-side view]



- | | | | | | |
|-----|---|------------|----|---|----------|
| 125 | ◆ | GND | 54 | ① | ECLIN |
| 32 | ◇ | ECLGND | 57 | ⊙ | ECLOUT |
| 76 | ⊕ | VDDGANTTGA | 0 | ⊗ | ECLBIDIR |
| 38 | ★ | VTTL | 26 | ① | TTLIN |
| 16 | ☆ | VECL | 37 | ⊙ | TTLOUT |
| 13 | ○ | NC | 20 | ⊗ | TTLBIDIR |
| 2 | ⊕ | ANALOG | | | |

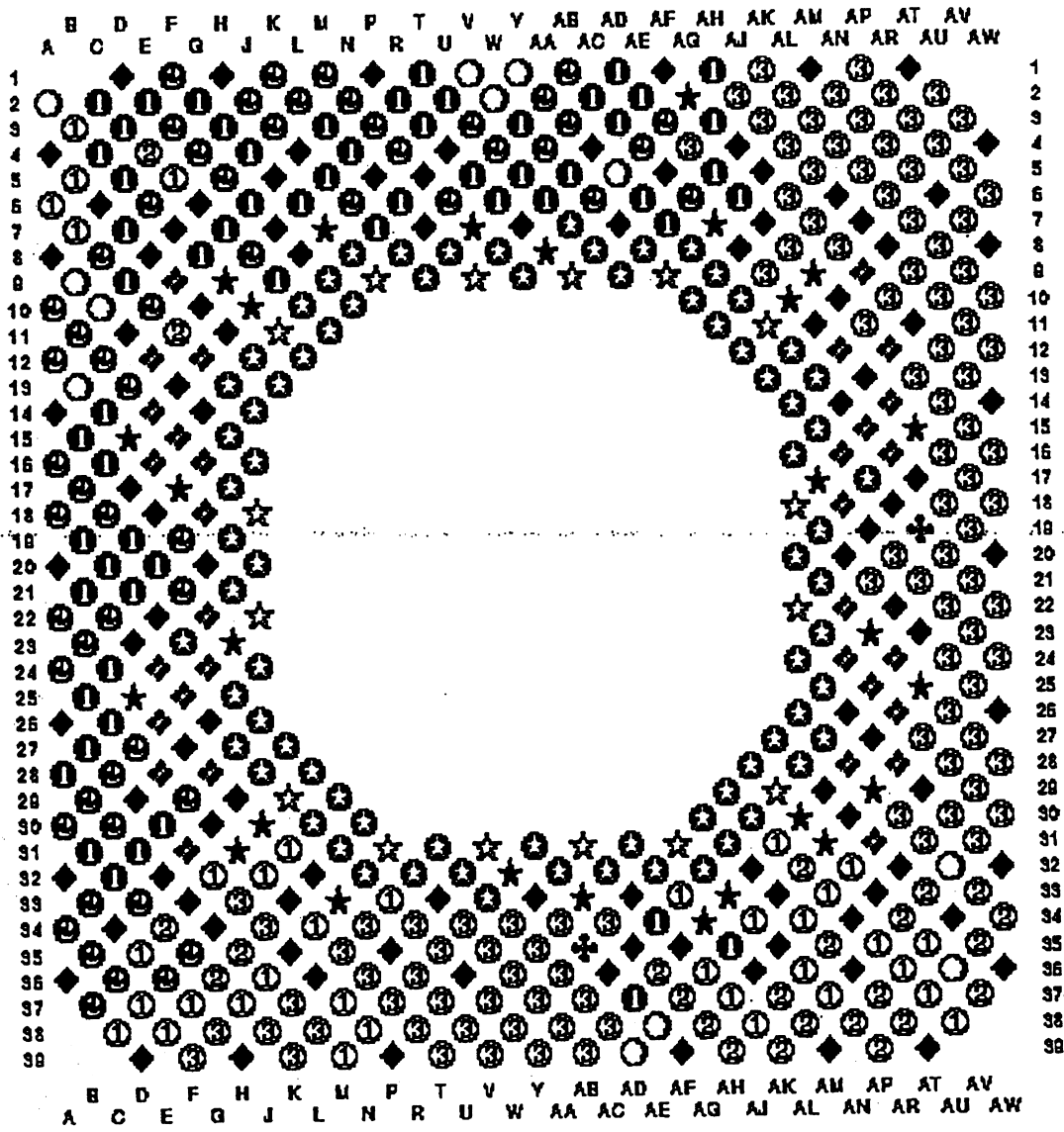
2) CCMC2

LGA557 PIN MAP FOR CCMC2 [probe-side view]



3) MAUI

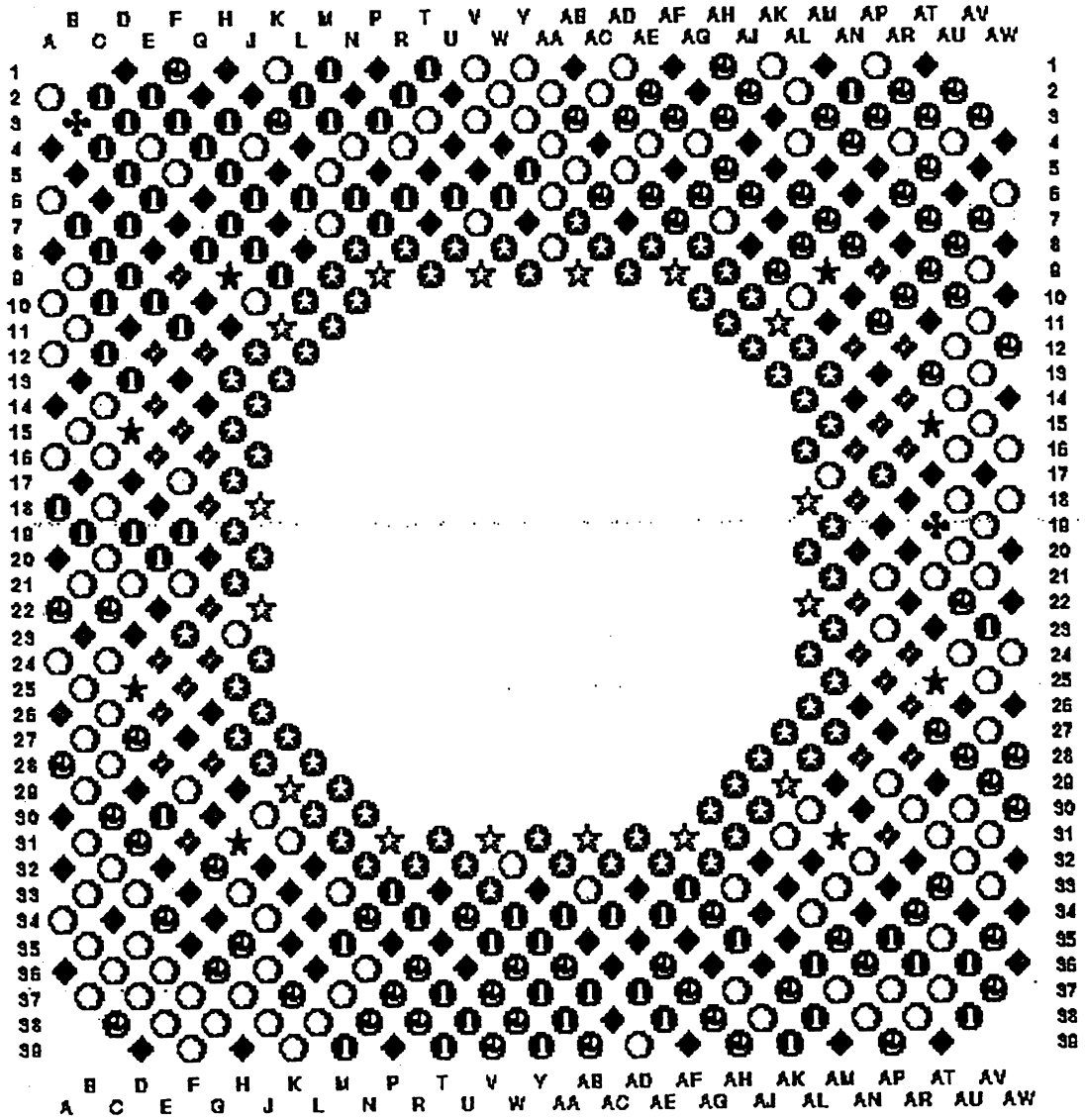
LGA557 PIN MAP FOR MAUI [probe-side view]



- | | | | | | |
|-----|---|------------|-----|---|----------|
| 106 | ◆ | GND | 61 | ① | ECLIN |
| 32 | ⊗ | ECLGND | 57 | ⊕ | ECLOUT |
| 76 | ⊙ | VDDGANTTGA | 0 | ⊖ | ECLBIDIR |
| 27 | ★ | VTTL | 36 | ① | TTLIN |
| 16 | ☆ | VECL | 24 | ② | TTLOUT |
| 12 | ○ | NC | 108 | ⊖ | TTLBIDIR |
| 2 | ⊕ | ANALOG | | | |

4) SCI2

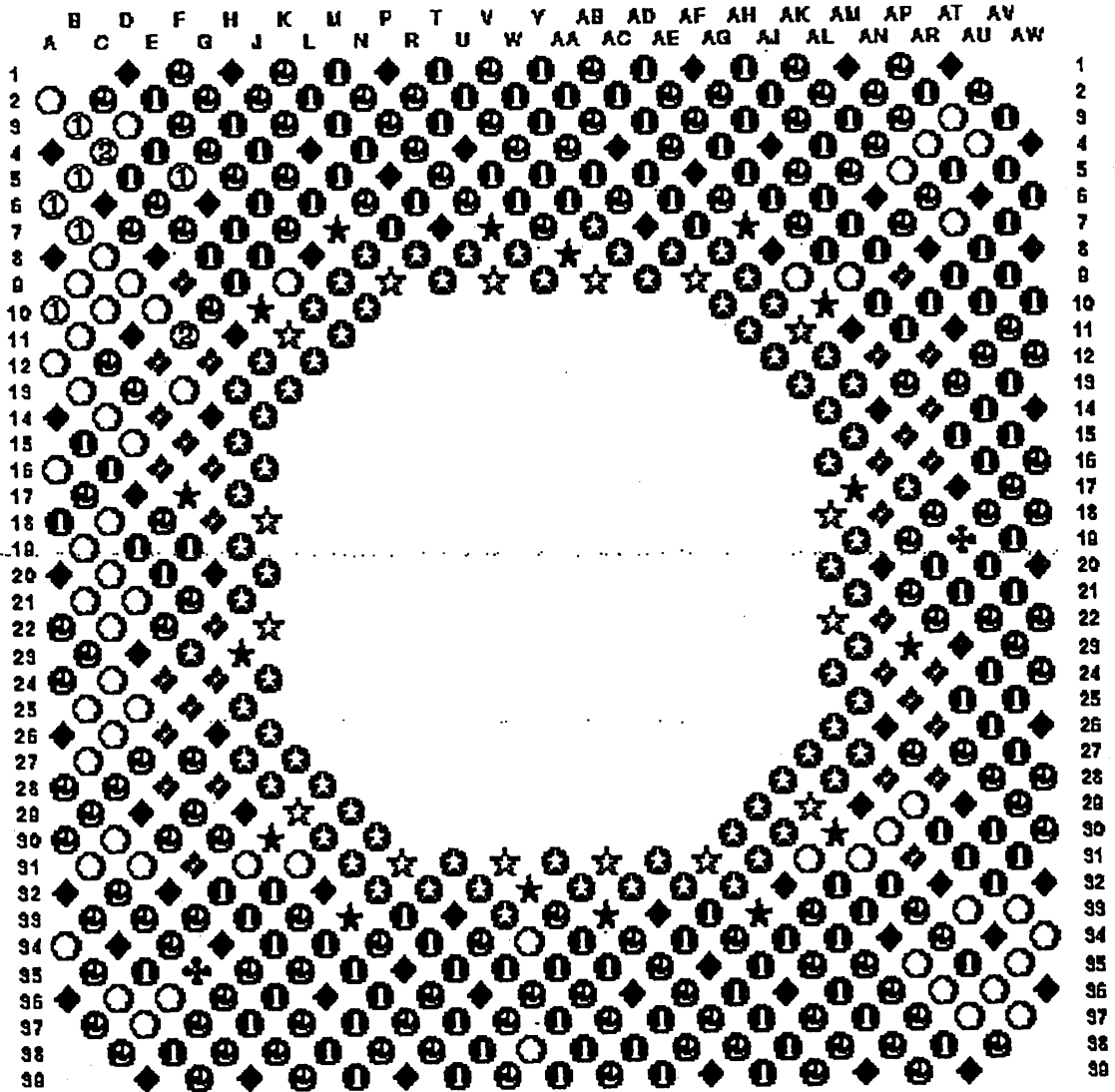
LGA557 PIN MAP FOR SCI2 [probe-side view]



- | | | | | | |
|-----|---|------------|----|---|----------|
| 197 | ◆ | GND | 79 | ① | ECLIN |
| 92 | ◆ | ECLGND | 81 | ② | ECLOUT |
| 76 | ⊗ | VDDGANTTGA | 0 | ③ | ECLBIDIR |
| 8 | ★ | VTTL | 0 | ④ | TTLIN |
| 16 | ☆ | VECL | 0 | ⑤ | TTLOUT |
| 192 | ○ | NC | 0 | ⑥ | TTLBIDIR |
| 2 | ⊕ | ANALOG | | | |

5) XBAR

LGA557 PIN MAP FOR XBAR [probe-side view]



76	◆	GND	134	①	ECLIN
92	◇	EOLGND	140	②	ECLOUT
76	⊗	VDDGA/TTGA	0	③	ECLBIDIR
16	★	VTTL	6	①	TTLIN
16	☆	VECL	2	②	TTLOUT
57	○	NC	0	③	TTLBIDIR
2	⊕	ANALOG			

Appendix • E Quick Reference Sheet

NAME	Description	where it lives	SPP-UX safe?
ccmu	Convex Configuration Utility	/spp/bin	yes*
cpptest	Test for the T-Chip modules	/spp/bin	no
cst	Convex Scan Test	/spp/bin	no*
diag_vers	displays the version of most files in the /spp/bin directory	/spp/bin	yes
do_reset	reset for the node(s)	/spp/bin	no
dump_all	display error register information.	/spp/scripts/cpa	no
dump_err	display error register information.	/spp/scripts/cpa	no
ecc_help	decode ECC error messages.	/spp/bin	yes
event_logger	background program that detects and reports events from the SPP.	/spp/bin	yes
fix_hung_inetd	kills and restarts all of the daemons in contact with the SPP on the TS.	/spp/bin	yes
hard_logger	interrogates the node(s) after a hard_error has occurred.	/spp/scripts	no
io_hang	interrogates the Landmarc to determine if one of the units are hung.	/spp/scripts	no
io_tc	landmarc test	/spp/bin	no
iod	I/O diagnostic shell	/spp/bin	no
kill_by_name	kill processes by name verses killing by pid.	/spp/bin	yes
load_eprom	program used to load NVRAM	/spp/bin	yes
load_mufw	Script which calls load_eprom used to load the MU firmware	/spp/scripts	yes
load_obp	Script which calls load_eprom used to load the OBP	/spp/scripts	yes
load_pl	Script which calls load_eprom used to load the MU primary loader	/spp/scripts	yes
mp1_map	Decodes memory information	/spp/bin	yes
node_info	Displays Node SN / IP address and MU firmware revision	/spp/bin	yes
node_ip_set	Sets the MU's IP address (if using the open Dart)	/spp/bin	yes
pce_util	Power, Clock and Environment Utility displays the status of the node	/spp/bin	yes*
restall	Restores ALL of the MU NVRAM firmware	/spp/scripts	yes
saveall	Saves ALL of the MU MVRAM firmware	/spp/scripts	yes
sn_cnsi	Gains control or spies the system's console	/spp/bin	yes
sppboot	Boots a node using the test station only	/spp/os	no
spring	Node memory and SCI ring tests	/spp/bin	no
sysreset	resets the node (linked to do_reset)	/spp/bin	no
unload_eprom	program used to save MU NVRAM firmware to a file	/spp/bin	yes
verify_cables	verifies the cabling of the SCI ring is correct	/spp/scripts	no
verify_rings	verifies the cabling of the SCI ring is correct	/spp/scripts	no
xconfig	GUI used to configure the MU parameters	/spp/bin	yes*

yes* = there are options that are not safe

no* = there are options that are safe

Appendix •F CCMU “gets” Definitions

INITWHAT

0x5880cfff

31	unknown
30	STARTPL
29	PLSELFTEST
28	PLCPUINIT
27	PLINITMEM
26	PLTESTMEM
25	PLLOADCODE
24	PLJUMPTOCODE
23	PLLOADOBP
22	PLSTATEDUMP
21	PLGETCACHE
20	PLERRENABLE
19	PLINITNODE
18	unknown
17	INITMEM
16	CLOCKGATE
15	MUERRENABLE
14	INITLAND
13	CHECKMEM
12	SIZEMEM
11	INITSLICE3
10	INITSLICE2
9	INITSLICE1
8	INITSLICE0
7	INITSCI3
6	INITSCI2
5	INITSCI1
4	INITSCIO
3	INITCXBAR
2	DORESET
1	PWRCLKCHECK
0	GO

RUNWHAT

0xf3ffffff

31	MB3
30	MB2
29	MB1
28	MB0
27	unknown
26	unknown
25	XBAR_Q
24	XBAR_S
23	SCI3
22	SCI2
21	SCI1
20	SCI0
19	CCMC3
18	CCMC2
17	CCMC1
16	CCMC0
15	TCHIP0
14	TCHIP1
13	TCHIP2
12	TCHIP3
11	TCHIP4
10	TCHIP5
9	TCHIP6
8	TCHIP7
7	CPA3
6	CPA2
5	CPA1
4	CPA0
3	IO1
2	IO0
1	MAUI
0	MU

NODEID

0x00000000

15 node 15
14 node 14
13 node 13
12 node 12
11 node 11
10 node 10
9 node 9
8 node 8
7 node 7
6 node 6
5 node 5
4 node 4
3 node 3
2 node 2
1 node 1
0 node 0

ASSERTRESET

0xffffffff

DEASSERTRESET

0x03fffffff

25	XBAR_Q
24	XBAR_S
23	SCI3
22	SCI2
21	SCI1
20	SCI0
19	CCMC3
18	CCMC2
17	CCMC1
16	CCMC0
15	TCHIP7
14	TCHIP6
13	TCHIP5
12	TCHIP4
11	TCHIP3
10	TCHIP2
9	TCHIP1
8	TCHIP0
7	CPA3
6	CPA2
5	CPA1
4	CPA0
3	IO1
2	IO0
1	MAUI
0	MU

USEMEMSIZE

0x00000005 - 024MB Boards
0x00000004 - 512MB Boards
0x00000003 - 256MB Boards
0x00000002 - 128MB Boards
0x00000001 - 64MB Boards
0x00000000 - 32MB Boards

USEBANKFIELD

0x00000004 - use all 4 banks on the MTVs

USECACHESIZE

0x00000000 - 0MB
0x00000001 - 4MB
0x00000002 - 8MB
0x00000003 - 16MB
0x00000004 - 32MB
0x00000005 - 64MB
0x00000006 - 128MB
0x00000007 - 265MB

TRACEMU

0x0000ffff - trace value for the MU's use

STATUSMU

0x00000080 - status value for the MU's use 80 = MU_STATUS_OK

APPLY_POWER

0x0000fff7

15	VTT_SCI_IO
14	IO
13	VEE_CLK
12	VHE_HP
11	VHC_HP
10	VCC_HP
9	VCC_MB3
8	VCC_MB2
7	VCC_MB1
6	VCC_MB0
5	VTT_SCI
4	VDD_SCI
3	unknown
2	VTT
1	VDL
0	VDD_GA

MU_CONFI

0x00000000

AVAIL_HW

0x80ff1fff

31	NODE
30	unknown
29	unknown
28	unknown
27	unknown
26	unknown
25	unknown
24	unknown
23	SCI3
22	SCI2
21	SCI1
20	SCI0
19	SLICE3
18	SLICE2
17	SLICE1
16	SLICE0

15	unknown
14	unknown
13	unknown
12	IO
11	MB3
10	MB2
9	MB1
8	MB0
7	TCHIP7
6	TCHIP6
5	TCHIP5
4	TCHIP4
3	TCHIP3
2	TCHIP2
1	TCHIP1
0	TCHIP0

AVAIL_NODES1

0x00000001 - node available

PRESENT_NODES1

0x00000001 - node present

HW_CONFIG

0x0000ff01

31	unknown
30	unknown
29	unknown
28	unknown
27	unknown
26	unknown
25	unknown
24	unknown
23	unknown
22	unknown
21	unknown
20	unknown
19	unknown
18	unknown
17	unknown
16	unknown
15	MEM_3_BANK_1
14	MEM_3_BANK_0
13	MEM_2_BANK_1
12	MEM_2_BANK_0
11	MEM_1_BANK_1
10	MEM_1_BANK_0
9	MEM_0_BANK_1
8	MEM_0_BANK_0
7	unknown
6	unknown
5	unknown
4	unknown
3	unknown
2	CLUSTER_MODE
1	CD_MODE
0	CMC_DARK

CLOCK_10MSEC

0x000f4240

APPLY_CLOCK

0x00000000 - Nominal Clock
0x00000004 - Upper Clock

INFO_MU

0x00000000

CPA_ERR_ENABLE

0x000007cf

31-12Non_Existent

11 Reserved

10 **SOFT_ERR_RES_EN** Enable ERROR response for Soft Errors9 **HPMCH0_EN** Enable HPMCH to processor 08 **LPMCH0_EN** Enable LPMCH to processor 07 **HPMCH1_EN** Enable HPMCH to processor 16 **LPMCH1_EN** Enable LPMCH to processor 15 **HARD_HPMCH_EN** Map HPMCH to Hard Errors4 **HARD_LPMCH_EN** Map LPMCH to Hard Errors3 **HARD_Q_EN** Q Input Port Hard Error Enable2 **HARD_S_EN** S Input Port Hard Error Enable1 **HARD_INT_EN** Internal Hard Error Conditions Enable0 **HARD_LOCK_FIRST** Lock hard_error CSR on first enabled error condition**CPA_PMON_LATENCY**

0x00000000

CPA_PMON_CONTROL

0x00000002

CMC_ERR_ENO

0xfcf7f3fff

- 31 RQI_PAR_ERR_XBAR request in, Parity error
- 30 RQI_SEQ_ERR_XBAR request in, Bad head-body-tail sequence
- 29 RQI_FMT_ERR_XBAR request in, Bad packet length
- 28 RSI_PAR_ERR_XBAR response in, Parity error
- 27 RSI_SEQ_ERR_XBAR response in, Bad head-body-tail sequence
- 26 RSI_FMT_ERR_XBAR response in, Bad packet length
- 25 NCI_PAR_ERR_Node chip incoming, Parity error
- 24 NCI_FMT_ERR_Node chip incoming, Bad packet format
- 23 SOCM_RS_ERR_Error response returned from node chip
- 22 BAD_ADDR_ERR_Illegal address
- 21 BAD_LAST_ERR_Bad last field during selected byte operation
- 20 BAD_CMD_ERR_Bad Major/Minor in request packet
- 19 VIRT_INDEX_ERR_Virtual index mismatch
- 18 MEM_ALIAS_ERR_Memory aliasing error (Bad BDT)
- 17 TRD_DIRTY_ERR_Instruction fetch to dirty line
- 16 NON_COH_ERR_Non-coherent access to coherent region from Xbar
- 15 SICM_NON_COH_ERR_Non-coherent access to coherent region from SCI
- 14 SICM_MISS_ERR_Cache miss from SCI
- 13 RQI_MISS_ERR_Retry/Unlock/copyout missed cache
- 12 NOT_OWNERR_ERR_Copyout from wrong processor
- 11 NOT_LOCKED_ERR_Retry/Unlock but lock bit not set (and should be)
- 10 RSQ_FULL_ERR_Resend Queue full
- 9 RC_DIRTY_ERR_Line already marked dirty in requesting processor's cache
- 8 SH_LEVEL_ERR_Sharing level error, Flush/Purge but line in use by SCI
- 7 CI_TRAP_ERR_Attempt to access another node (enable on single node systems only)
- 6 RFRSH_ERR_Refresh error
- 5 SBIT_ECC_ERR_Single-bit ECC error
- 4 MBIT_ECC_ERR_Multi-bit ECC error
- 3 MULT_ECC_ERR_Multiple ECC errors have occurred
- 2 RSI_BAD_RSP_ERR_Bad response received by RSI
- 1 UNREC_TAG_ERR_Unrecoverable tag error, Multi-bit ECC error while reading tags for RSI or SOCM_RQ
- 0 FLUSH_RP_ERR_Multi-bit ECC error while making flush or flush request to SCI

CMC_ERR_EN1

0x00000000

63-32 unused

0x00000000 MSIZE032M memory boards node 0
0x10000000 MSIZE064M memory boards node 0
0x20000000 MSIZE0128M memory boards node 0
0x30000000 MSIZE0256M memory boards node 0
0x40000000 MSIZE0512M memory boards node 0
0x50000000 MSIZE01024M memory boards node 0

0x22220000 MSIZE0128M memory boards node 0 1 2 3
0x22222222 MSIZE1128M memory boards node 8 9 10 11 12 13 14 15

Appendix •G MU Status Numbers

Mu Status Codes for V7.2.1

* DART status values	

MU_STATUS_DART_BUSY	(0x0200 + MU_STATUS_BUSY)
MU_STATUS_DART_UNAVAILABLE	0x0293
MU_STATUS_DART_BAD_NODE	0x0292
MU_STATUS_DART_TIMEOUT	0x0291
MU_STATUS_DART_TRUNCATED	0x0290

* SONIC status values	

MU_STATUS_SONIC_SEND_QUEUE	0x0103
MU_STATUS_SONIC_RECEIVED	0x0102
MU_STATUS_SONIC_UNUSED	0x0101

* generic status values	

MU_STATUS_BAD_ADDR	0x00ff
MU_STATUS_BAD_SIZE	0x00fe
MU_STATUS_ZERO_SIZE	0x00fd
MU_STATUS_BAD_TYPE	0x00fc
MU_STATUS_SHORT_MSG	0x00fb
MU_STATUS_BUSY	0x00fa
MU_STATUS_PANIC	0x00f9
MU_STATUS_SUBTEST_FAIL	0x00f8
MU_STATUS_BUS_ERROR	0x00f7
MU_STATUS_NO_BUS_ERROR	0x00f6
MU_STATUS_LIMIT_EXCEEDED	0x00f5
MU_STATUS_REQ_IGNORED	0x00f4

* subtest status values	

MU_STATUS_ST_COP_ERROR	0x00d6
MU_STATUS_ST_TIMEOUT	0x00d5
MU_STATUS_ST_BYTE_SELECT_ERROR	0x00d4
MU_STATUS_ST_ACCESS_ERROR	0x00d3
MU_STATUS_ST_PARITY_ERROR	0x00d2
MU_STATUS_ST_ADDRESS_ERROR	0x00d1
MU_STATUS_ST_PATTERN_ERROR	0x00d0

```

*-----
* miscellaneous status values
*-----
MU_STATUS_OK                0x0080
MU_STATUS_WRITE             0x0004
MU_STATUS_READ              0x0003
MU_STATUS_IN_USE           0x0002
MU_STATUS_UNUSED           0x0001

*-----
* CXRing configuration status values
*-----
MU_STATUS_CXC_TIMEOUT_2    0x1091
MU_STATUS_CXC_TIMEOUT_1    0x1090

*-----
* console status values
*-----
MU_STATUS_CONSOLE_MAX_OPEN  0x0f94
MU_STATUS_CONSOLE_WR_OVERRUN 0x0f93
MU_STATUS_CONSOLE_OPEN_ID   0x0f92
MU_STATUS_CONSOLE_RD_OVERRUN 0x0f91
MU_STATUS_CONSOLE_BAD_ID    0x0f90

*-----
* memory allocation status values
*-----
MU_STATUS_MALLOC_NO_MEM     0x0e90

*-----
* trace status values
*-----
MU_STATUS_TRACE_BAD_PAGE    0x0d91
MU_STATUS_TRACE_BAD_TYPE    0x0d90

*-----
* reset status values
*-----
MU_STATUS_RESET_BUSY        (0x0c00 + MU_STATUS_BUSY)
MU_STATUS_RESET_BAD_LEVEL   0x0c90

*-----
* COP status values
*-----
MU_STATUS_COP_BUSY          (0x0b00 + MU_STATUS_BUSY)
MU_STATUS_COP_BAD_ID        0x0b91
MU_STATUS_COP_TIMEOUT       0x0b90

```

* clock control status values

MU_STATUS_CLOCKS_SCI_ENABLED	0x0a97
MU_STATUS_CLOCKS_BAD_CSB_TYPE	0x0a96
MU_STATUS_CLOCKS_MODE_SET	0x0a95
MU_STATUS_CLOCKS_BAD_MODE	0x0a94
MU_STATUS_CLOCKS_SOURCE_SET	0x0a93
MU_STATUS_CLOCKS_BAD_SOURCE	0x0a92
MU_STATUS_CLOCKS_ENABLED	0x0a91
MU_STATUS_CLOCKS_DISABLED	0x0a90

* power and environment status values

MU_STATUS_PNE_BUSY	(0x0900 + MU_STATUS_BUY)
MU_STATUS_PNE_PWR_BAD_CONFIG	0x09b1
MU_STATUS_PNE_PWR_VDL_AND_TCHIPS	0x09b0
MU_STATUS_PNE_WEAK_48V	0x099a
MU_STATUS_PNE_PWR_SHUTDOWN_HI	0x0999
MU_STATUS_PNE_PWR_SHUTDOWN_LO	0x0998
MU_STATUS_PNE_PWR_NO_SUPPLY	0x0997
MU_STATUS_PNE_PWR_OFF_ABORT	0x0996
MU_STATUS_PNE_PWR_ADJUST_TIMEOUT	0x0995
MU_STATUS_PNE_PWR_BAD_CALIBRATE	0x0994
MU_STATUS_PNE_PWR_MEASURE_TIMEOUT	0x0993
MU_STATUS_PNE_PWR_ON_TIMEOUT	0x0992
MU_STATUS_PNE_TRIM_TIMEOUT	0x0991
MU_STATUS_PNE_BAD_CHANNEL	0x0990

* script status values

MU_STATUS_SCRIPT_BUSY	(0x0800 + MU_STATUS_BUSY)
MU_STATUS_SCRIPT_BAD_OPCODE	0x0892
MU_STATUS_SCRIPT_STACK_UNDERFLOW	0x0891
MU_STATUS_SCRIPT_STACK_OVERFLOW	0x0890

* configuration status values

MU_STATUS_CONFIG_BAD_VALUE	0x0792
MU_STATUS_NODE_ID_CONFLICT	0x0791
MU_STATUS_CONFIG_BAD_NODE_ID	0x0790

* NVRAM status values

MU_STATUS_NVRAM_BUSY	(0x0600 + MU_STATUS_BUSY)
MU_STATUS_NVRAM_ZERO_CHECKSUM	0x0696
MU_STATUS_NVRAM_BAD_SL_CHECKSUM	0x0695
MU_STATUS_NVRAM_BAD_PL_CHECKSUM	0x0694
MU_STATUS_NVRAM_BAD_FW_CHECKSUM	0x0693
MU_STATUS_NVRAM_BAD_KEY	0x0692
MU_STATUS_NVRAM_FULL	0x0691

MU_STATUS_NVRAM_VAR_NOT_FOUND	0x0690

* EEPROM load status values	

MU_STATUS_EEPROM_BUSY	(0x0500 + MU_STATUS_BUSY)
MU_STATUS_EEPROM_WRITE_FAILED	0x0592
MU_STATUS_EEPROM_NOT_IN_USE	0x0591
MU_STATUS_EEPROM_IN_USE	0x0590

* scan status values	

MU_STATUS_SCAN_BUSY	(0x0400 + MU_STATUS_BUSY)
MU_STATUS_SCAN_BAD_ID	0x0496
MU_STATUS_SCAN_IN_USE	0x0495
MU_STATUS_SCAN_NOT_OWNER	0x0494
MU_STATUS_SCAN_ERROR	0x0493
MU_STATUS_SCAN_BAD_RING	0x0492
MU_STATUS_SCAN_BAD_END_STATE	0x0491
MU_STATUS_SCAN_BAD_COMMAND	0x0490
MU_STATUS_SCAN_RW_SHORT	0x0433
MU_STATUS_SCAN_RW	0x0432
MU_STATUS_SCAN_RW_START	0x0431
MU_STATUS_SCAN_WRITE	0x0421
MU_STATUS_SCAN_READ	0x0412
MU_STATUS_SCAN_READ_START	0x0411
MU_STATUS_SCAN_START	0x0401

* MAUI status values	

MU_STATUS_MAUI_BUSY	(0x0300 + MU_STATUS_BUSY)
MU_STATUS_MAUI_REMOTE_MASK	0x1000
MU_STATUS_MAUI_UNAVAILABLE	0x03a6
MU_STATUS_MAUI_MSG_TIMEOUT	0x03a5
MU_STATUS_MAUI_MSG_TOO_LARGE	0x03a4
MU_STATUS_MAUI_NO_STATUS	0x03a3
MU_STATUS_MAUI_BAD_CHANNEL	0x03a2
MU_STATUS_MAUI_CHANNEL_IN_USE	0x03a1
MU_STATUS_MAUI_NO_CHANNELS	0x03a0
MU_STATUS_MAUI_TIMEOUT	0x0391
MU_STATUS_MAUI_CXBAR_ERROR	0x0390
MU_STATUS_MAUI_AWAIT_COH_FLUSH_G_RES	0x0320
MU_STATUS_MAUI_INCOMING_MSG	0x0310
MU_STATUS_MAUI_AWAIT_SCRUB_TAG_RES	0x0307
MU_STATUS_MAUI_AWAIT_SCRUB_DATA_RES	0x0306
MU_STATUS_MAUI_AWAIT_CXBAR_RES	0x0305
MU_STATUS_MAUI_AWAIT_NWR_RES	0x0304
MU_STATUS_MAUI_AWAIT_NRD_RES_64	0x0303
MU_STATUS_MAUI_AWAIT_NRD_RES_SB	0x0302
MU_STATUS_MAUI_UNUSED	0x0301
