CONVEX

Exemplar Series Systems
Hardware Maintenance Manual

Version 8.2

Compiled and Edited
by Randy Zeid

September 14, 1995

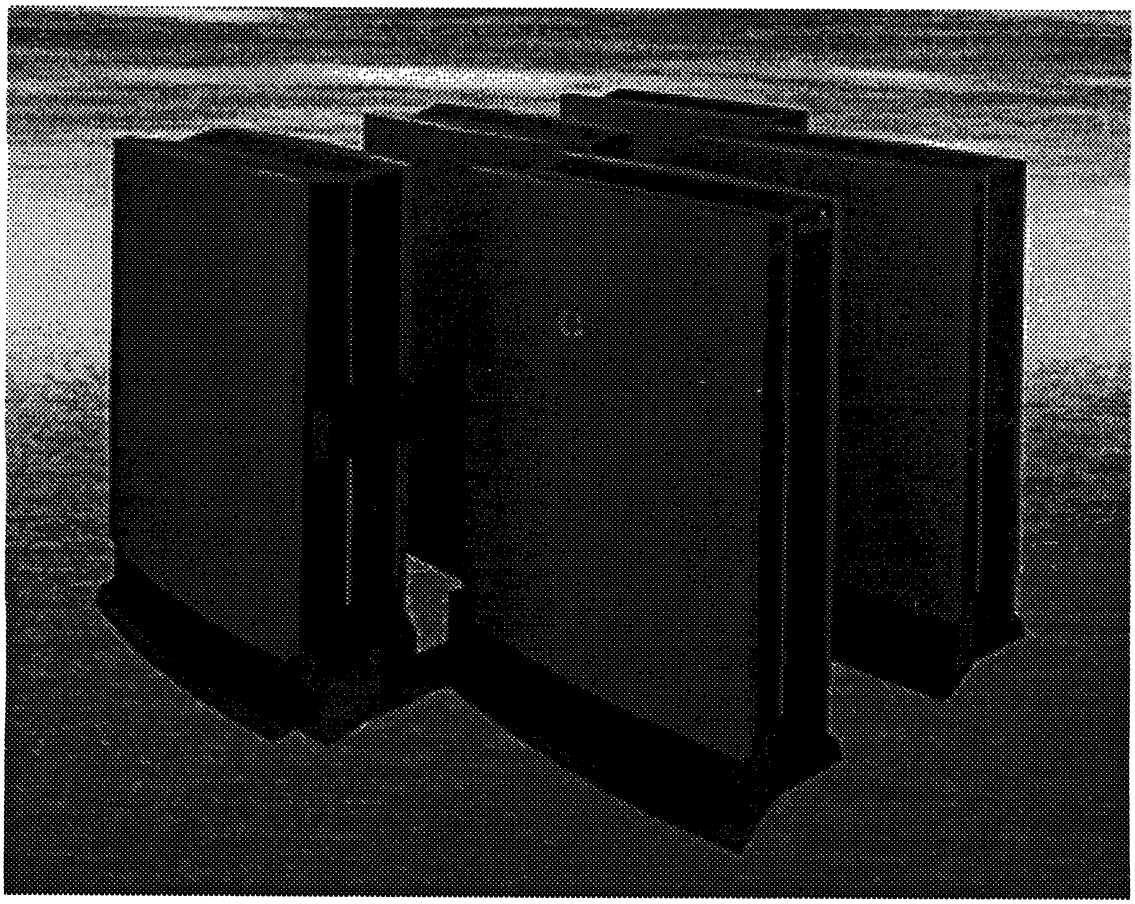


Table of Contents

Section 1 Exemplar System Overview

Basic Definitions **1**

Test Station 1

DaRT 1

Node 2

Node Chassis 2

Slice 2

CSB/ ASB 3

Cabinet 3

Complex 3

Power System Descriptions **5**

Power Brick 5

Power Brick Board 5

Power Concentrator Card 5

Power Chassis 5

Logic Systems **6**

Exemplar Processors 6

MU 6

MTV 7

I/O Interface Cards 7

CPA 7

CCMC 8

XBAR 8

MAUI 8

SCI 8

Misc. Definitions **10**

MOD 10

Front Panel 10
OBP 10
SPP-UX 11
PVM 11
GSM 11

Section 2 Standard Configurations

Configurations **13**
XA Single Node **14**
XA 2 Node Configuration **15**
CD Single Node **17**
XA Single Node Multi-I/O **18**
CD Multi-IO **19**
XA - Max Configuration **21**
Standard 6 Node Configuration **23**

Section 3 The DaRT Bus

DaRT bus **25**
DaRT and /etc/host **26**
DaRT Cable 27

Section 4 Front Panel LCD

Front Panel Display **29**
Typical Front Panel Messages **30**

Section 5 CSB/ASB Level Functional units

Introduction **31**
CSB4 Physical Layout 31
ASB Physical Layout **35**
Agent Gate Array **38**
Agent Hard Errors 40
CCMC Gate Array **41**
CCMC Functional Block diagram 43
CCMC Hard Errors 44

Xbar Gate Arrays 45
MAUI Gate Array **47**
MAUI Hard Errors 49
SCI Gate Array **50**

Section 6 Boards

T-Chip Module - SPP1000 ..... **53**
T' (T Prime) Chip Module **53**
Mpp Utility board **56**
MU EEPROM and DRAM 57
Memory Tag Vortex Logic Card **59**
Landmarc/ Avalanche **63**

Section 7 The Power System

Introduction **65**
Power Sequencing (ON/OFF) **66**
Power Brick Descriptions **68**
Power Board 0 68
Power Board 1 68
Power Board 2 68
200 Series Power bricks 69
APB Avalon Power Board 70
Power brick locations **71**
Power Chassis Wiring **72**
Power Chassis Front Panel **73**
Power Chassis Concentrator Card **74**

Section 8 MEMORY

Physical memory divisions **75**

Section 9 Internal Communications

The Physical data path **79**
Packet Types **81**
Request Packets 81

Response (Send) Packets 82
Packet Structures **84**

Section 10 Meaningful Optical Display

Introduction **89**
The MOD light controller **89**
MOD light cable routing **90**
MOD Board Strapping **91**
Part Numbers **92**

Section 11 Gate Array Removal and Replacement

Field Gate Array Replacement **93**
Removal and Replacement Parts **93**
Cleaning **95**
Maintenance **95**
Orientation **96**

Section 12 I/O

Boards **97**
Peripheral Configurations **97**
XA **98**
High Performance 98
High Capacity 100
Power system 102
Slot Addresses 103
CD Configuration **104**
Disk Drives - Strapping **105**
DEC (2 GIG) 105
Segate (4 GIG) 106
Dat Drive 108
Ancot Card 109
IO Address Reference Sheet (nvalias format) **110**
OBP Setup Parms for IO Devices (mkmap) **112**
Dat Drive Examples 112
FDDI Example 112

Section 13 Test Station

The file system tree **113**
Important Test Station files **114**
Useful Test Station Commands **115**
Backup - In SPP-UX 115
Backup - HP UX (test station files) 115
Restore 116
Creating a Bootable Recovery Tape - HP-UX 116
Software Install 117
Test Station Configuration 117
Boot single on the Test Station 117
Boot SPP-UX From the Dat Drive 117

Section 14 Diagnostics & Utilities

Test Station SW Introduction **119**
Software/ Firmware Versions. **121**
MU "core" 121
Mu_firmware versions. 122
Diag_vers Command 123
SPP-UX Versions. 123
CCMU Camelot Configuration Management Utility **124**
The CCMU path diagram **126**
Important CCMU Parameters **127**
Turning off OBP **128**
Configuration Errors **129**
Correct From Scratch - Using CCMU 129
Correct From File 130
Using Saveall and Restall 130
Xconfig All **131**
Cputest: T-CHIP Diagnostic **138**
Cputest: T'-CHIP Diagnostic **138**
CST - Convex Scan Test **139**
CST Connectivity Failure Message 149

Do_reset **150**
Sysreset Failure Message 151
Ecc_help - ECC Error Decode Utility **152**
Event_logger MPP Test Station Event Logger **154**
Event_log **155**
iod - I/O Diagnostic shell **156**
io_tc **157**
kill_by_name **159**
load_eprom **160**
Reloading the MU firmware **161**
Reloading the Primary Loader **161**
Reloading the Secondary Loader **162**
New Versions of the load Commands **162**
Saveall - Saving NVRAM Contents. **163**
Restall - Restoring NVM contents. **164**
mp1_map - Address Decode Tool **165**
Node_info **168**
Pce_util - SPP Power, Clock & Environment Utility **169**
pce_util output 171
Sn_cnsl **172**
SPP Diagnostic Shell **173**
Sppring CxRing and Distributed Memory Test **174**
Class 1 Access Verification subtests 174
Class 2 Distributed Memory (Tag) Subtests 174
Class 3 Distributed Memory (Data) Subtests 174
Class 4 Non-Coherent Subtests 175
Class 5 Download Verification and EIR Check Subtests 175
Class 6 Intranode Coherency Subtests 175
Class 7 CxRing Internode Coherency Subtests 175
Class 8 Message Subtest 175
Class 9 SCI Stress Subtests 176
Class 10 Coherency Stress Subtests 176
Class 11 SIMM Quick Sanity Subtests 176
Class 12 Tag SIMM quick check 176
Class 13 Data SIMM quick check 176

MU power on Diagnostic **177**
node_ip_set **180**
io_hang **181**
dump_err **182**
dump_all **183**
verify_cables **184**
verify_rings **186**

Section 15 SPP-UX

Boot Command **187**
Network Files **188**
Disk Administration Under SPP-UX **188**
Unmap Disk **189**
MAP Disk to device **189**
PARTITION Disk **189**
NEWFS **190**
SPP-OS FYI **190**
Remote console commands 190
Passwords 191
OBP Open Boot Prom Architecture **191**
OBP Commands 192
Example of a normal OBP banner. 193
OBP standard parameters - defaults are GSM 193
OBP PVM-CTI VS GSM 194
Important Boot Files 195
SPP-UX version checks 195

Section 16 Multi-Node Operation & Setup

Technical Information **197**
SCI RING Connection **198**
SCI diagnostics 200
Multi-cabinet connection & Power sequencing 200
Current Multi-node HW Requirements **201**
TOC cable Connections **202**

Section 17 Procedures and Trouble Shooting Aids

CD Installation Procedure **203**
Identifying When the IO Board is Hung **207**
Autoconfig network on boot **208**
Initial install network setup **209**
/etc/netlinkrc 209
Determining the Source of a Hard Error **210**
SPP-UX Hang and Panics Procedure **211**
Hard Error Debug (Agent hard error check) **215**

APPENDICES

Appendix A **: Class Documentation list 217**

Appendix B **: Tools..... 218**

Appendix C **: FRU List..... 219**

Appendix D **: Gatearray pin-out 221**

Appendix E **: Quick Reference Sheet..... 226**

Appendix F **: CCMU "gets" Definitions 227**

Appendix G **: MU Status Numbers..... 238**

List of Figures

Section 1 Exemplar System Overview
Physical layout of the CSB 4
CSB block diagram 9

Section 2 Standard Configurations
XA - Single Node Configuration 14
XA - 2 Node Configuration 15
XA Leveling Feet 16
CD Single Node configuration 17
CD W/Test Station and IO chassis 17
XA Single Node Multi-I/O configuration 18
CD - Multi-IO 19
CD - I/O Chassis and cover 20
16 Node Configuration 22
XA 6 Node Configuration 23

Section 3 The DaRT Bus
Dart bus /etc/host entries 26
The DaRT bus cable 28

Section 4 Front Panel LCD
Front Panel Messages 30

Section 5 CSB/ASB Level Functional units
CSB Layout Z-Side 32
CSB U-Side Layout 33
CSB Data Paths 34
ASB Physical Layout 35
ASB U-SIDE 36
ASB Data Paths 37
Agent Gate Array Block Diagram 39

CCMC Memory Interface Block diagram 42
CCMC Functional Block diagram 43
Xbar Block Diagram 46
MAUI Block Diagram 48
SCI Block Diagram 51

Section 6 Boards
T-Chip Module Block Diagram 55
MU Block Diagram 56
MU LEDs 57
MU Memory Allocation Map 58
MTV Block diagram 60
MTV Physical Layout 61
Memory configuration table 62
Landmarc Block Diagram 64

Section 7 The Power System
Power Subsystem Diagram 67
Power Brick Locations 71
Power Supply Wiring 72
Power Chassis Front Panel 73
Power Concentrator Card 74

Section 8 MEMORY
Node Physical Address Space Partitioning 77
Physical Memory Addressing with Ring Bits 78

Section 9 Internal Communications
Packet Word Format 85
Packet word formats (Continued) 86

Section 10 Meaningful Optical Display
Mod light power and data cable routing 90
MOD Board Strapping Diagram 91

Section 11 Gate Array Removal and Replacement

Gate Array Removal and Replacement 94
Gatearray and Socket orientation 96

Section 12 I/O

The High Performance Peripheral chassis 98
The High Performance Block Diagram 99
The High Capacity Configuration 100
The High Capacity Block diagram 101
Peripheral cabinet power subsystem 102
High Density SCSI ID's 103
CD cable diagram 104
Disk Drive Strapping (DEC DSP3210W 2) 105
DAT Drive strapping 108
Differential to Single Ended Converter Card 109
IO Address Figure 111

Section 13 Test Station

Section 14 Diagnostics & Utilities

Test Station Software Overview 120
CCMU diagram 126
Xconfig Display 131
Xconfig Config Pop-up Menu 132
Xconfig Parms Pop-up Menu 133
Xconfig Parms/Clocks Pop-up Menu 134
Xconfig Parms/Memory Options Pop-up Window 135
Xconfig Help Pop-up Window 136
Xconfig Help/Color Key Pop-up Window 137
CST GUI Display 140
CST Scan Ring Partitioning 141
CST Files Menu 142
CST Options Menu 143
CST Power Menu 144
CST Clocks Menu 145

CST Details Menu 146
CST Misc. Menu 147
CST Help Screen 148

Section 15 SPP-UX

Section 16 Multi-Node Operation & Setup
SCI XA Cabling Diagram 198
Node XA SCI Cabling Diagram 199
SCI CD Cabling Diagram 200
TOC cable connection diagram 202

Section 17 Procedures and Trouble Shooting Aids

List of Tables

Section 5 CSB/ASB Level Functional units

AGENT

Internal errors 40

External errors 40

CCMC

Hard Errors 44

MAUI

Internal Errors 49

External Errors 49

Section 6 Boards

I/O Boards 63

Section 9 Internal Communications

Typical packet formats (request packets) 87

Exemplar System Overview 1

1.1 Basic Definitions

In this section the "basic" building blocks of the SPP-1000/ 1200 series systems will be covered.

1.1.1 Test Station

- The **test station** is a dedicated work station connected to the SPP system via an ethernet "like" 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:
    - one CPU
    - one memory board
    - one backplane with the associated support gate arrays
    - one MU (utility board)
    - one ~~MAUI~~ (I/O board)
    - power brick boards (spp1000 - 3, spp1200 - 1).

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

1.1.4 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.

1.1.5 Slice

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

1.1.6 CSB/ ASB

- The **C**amelot **S**ystem **B**oard 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 4** is a physical layout of the CSB.
  - **Figure 1-2 on page 9** is a block diagram of the CSB.

- In the spp1200 the system board is called the **ASB,** **A**valon **S**ystem **B**oard.

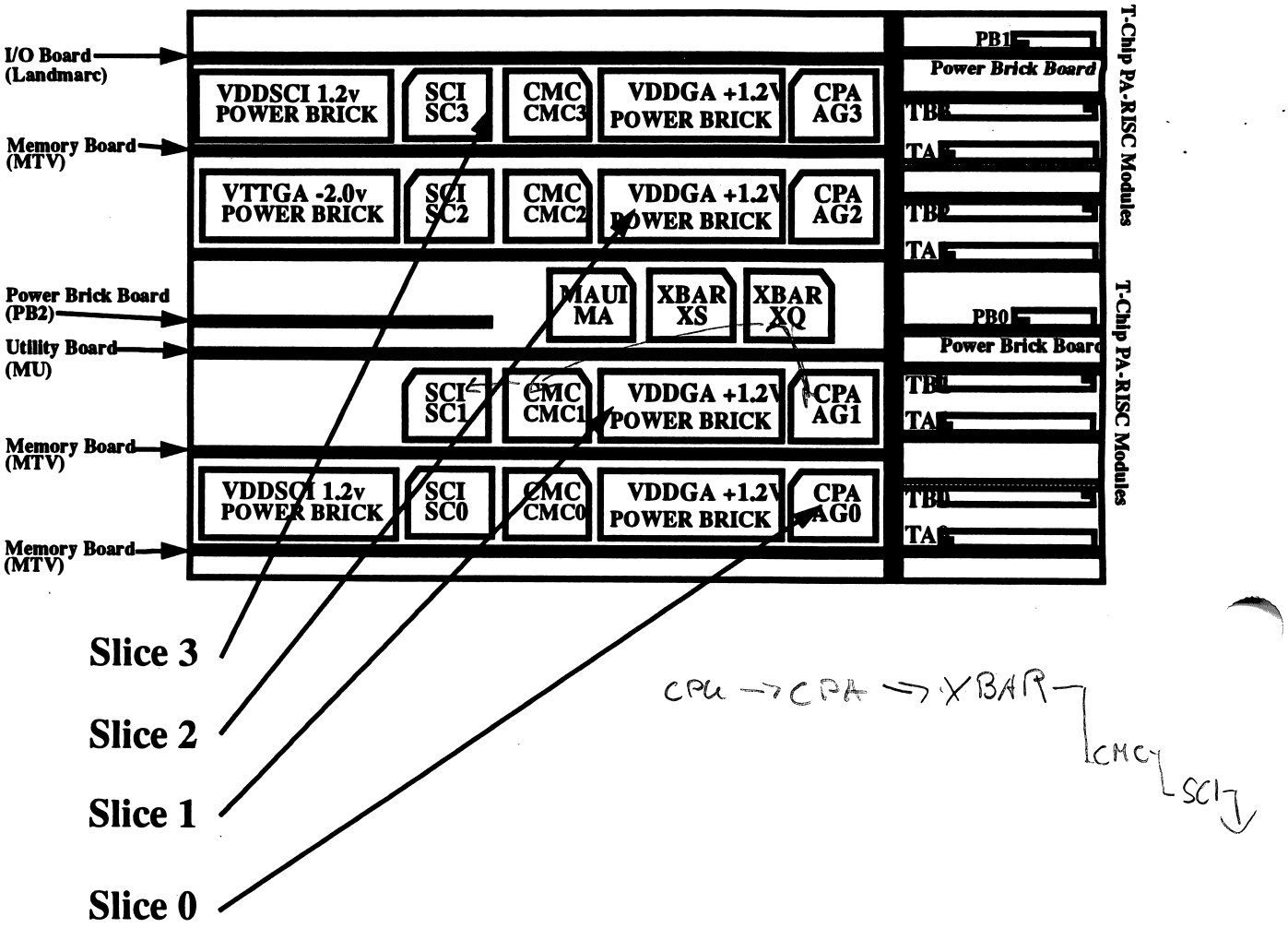
1.1.7 Cabinet

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

1.1.8 Complex

- A **Complex** is one or more Nodes connected together as a single system.

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 SPP-1000.
  - A power brick is used to convert a DC voltage to a higher current but lower DC voltage.
  - In all cases the power bricks take 48VDC as their input voltage.
  - All of the system power is derived by a single 48VDC power source converted to the required voltages via the power bricks.

1.2.2 Power Brick Board

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

1.2.3 Power Concentrator Card

- A small circuit board used to monitor and control power on SSP XA model.

1.2.4 Power Chassis

- AC input voltage enters the system here,
  - where AC power is conditioned
  - and connected to the system via the **main breaker.**

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 (CSB) can have up to 8 modules - 2 per slice.
  - The **T-Chip** uses the HP PA-RISC (7100 series) 99MHZ processor with 1M of Data and Instruction cache.

- **T' (T-prime)** is the processor module used in the SPP1200.
  - Can run at clock speeds up to 120MHZ
  - PA RISC 7200 series.
  - Cache size - 256K

1.3.2 MU

Master wurde zuerst hier das SPP-System vorgestellt

- The **MU** (Mpp Utility card) is the SPU-like board for the SPP.
  - It provides the testation interface as well as the system environmental monitoring.
  - Other run-time functions are supplied.
  - There is only 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.
  - Currently, only the 4 SBUS slot are usable.
  - SIOP I and SIOP I'
- **Avalanche** - a newer version of the landmarc which provides for up to 8 sbus controllers.
  - SIOP II

1.3.5 CPA

- **CPu Agent** gate array.
  - It provides the interface between the T-Chip modules and the rest of the SPP.
  - It also provides the arbitration between the 2 CPUs in a slice.
  - A node can have up to 4 CPA's, 1 per slice.
  - The CPA lives directly on the CSB.

1.3.6 CCMC

- The **Convex Coherent Memory Chip.**
  - 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 even though it is physically spread out over the node.

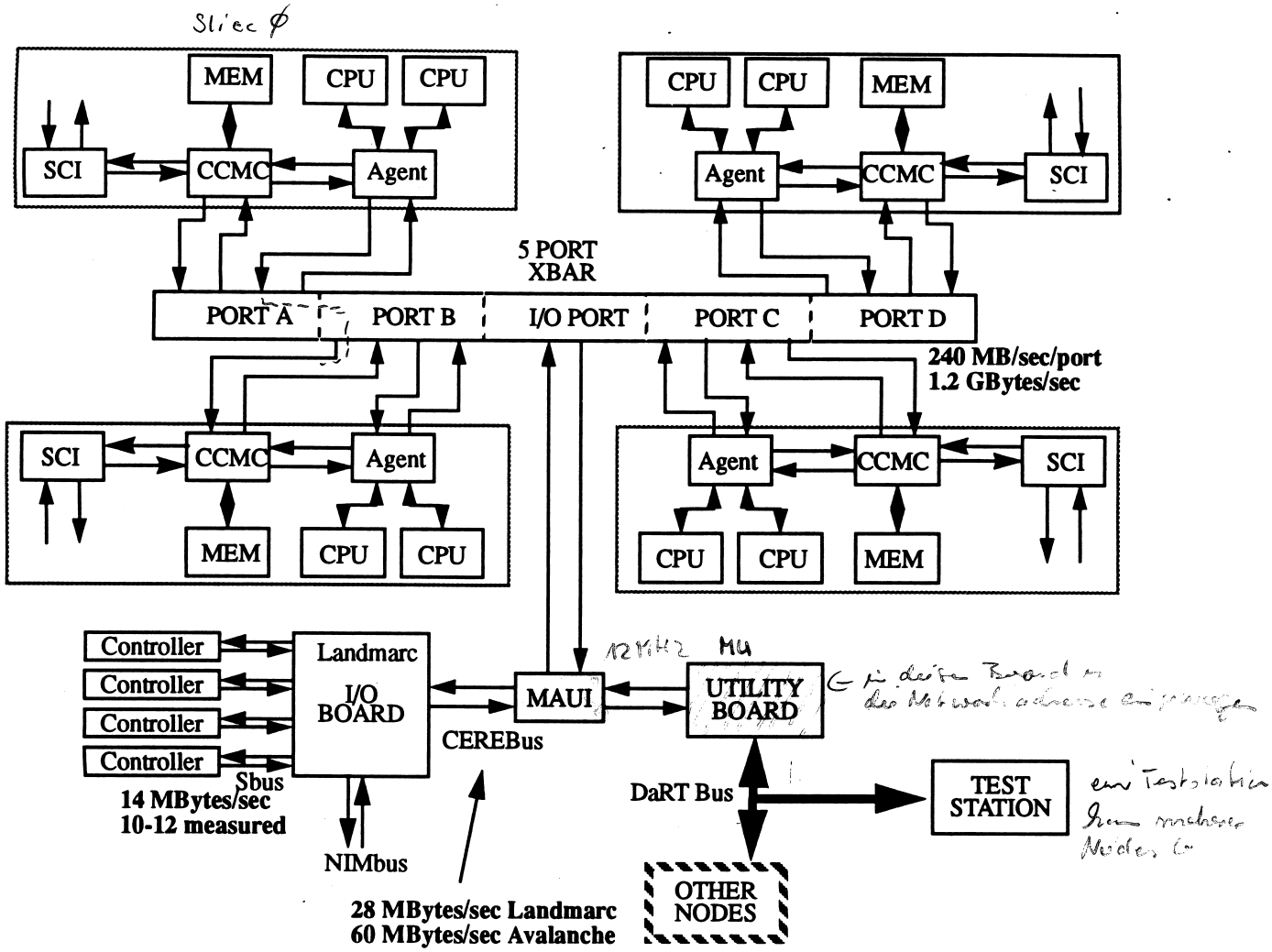
1.3.8 MAUI

- **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 I/O card 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 though this high-speed link that global memory 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 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 node.

1.4.4 SPP-UX

- **SPP-UX** is the name of the Operating System used on the SPP-1000.
  - It is different in many ways from ConvexOS.
  - It has been patterned after HP-UX and only differs from it where the hardware forces it to be different.

1.4.5 PVM (Software)

- 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.
  - It enables idle cpus to work the application as resources become available.

1.4.6 GSM

- The Global Shared Memory is a section on the 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

Standard Configurations 2

2.1 Configurations

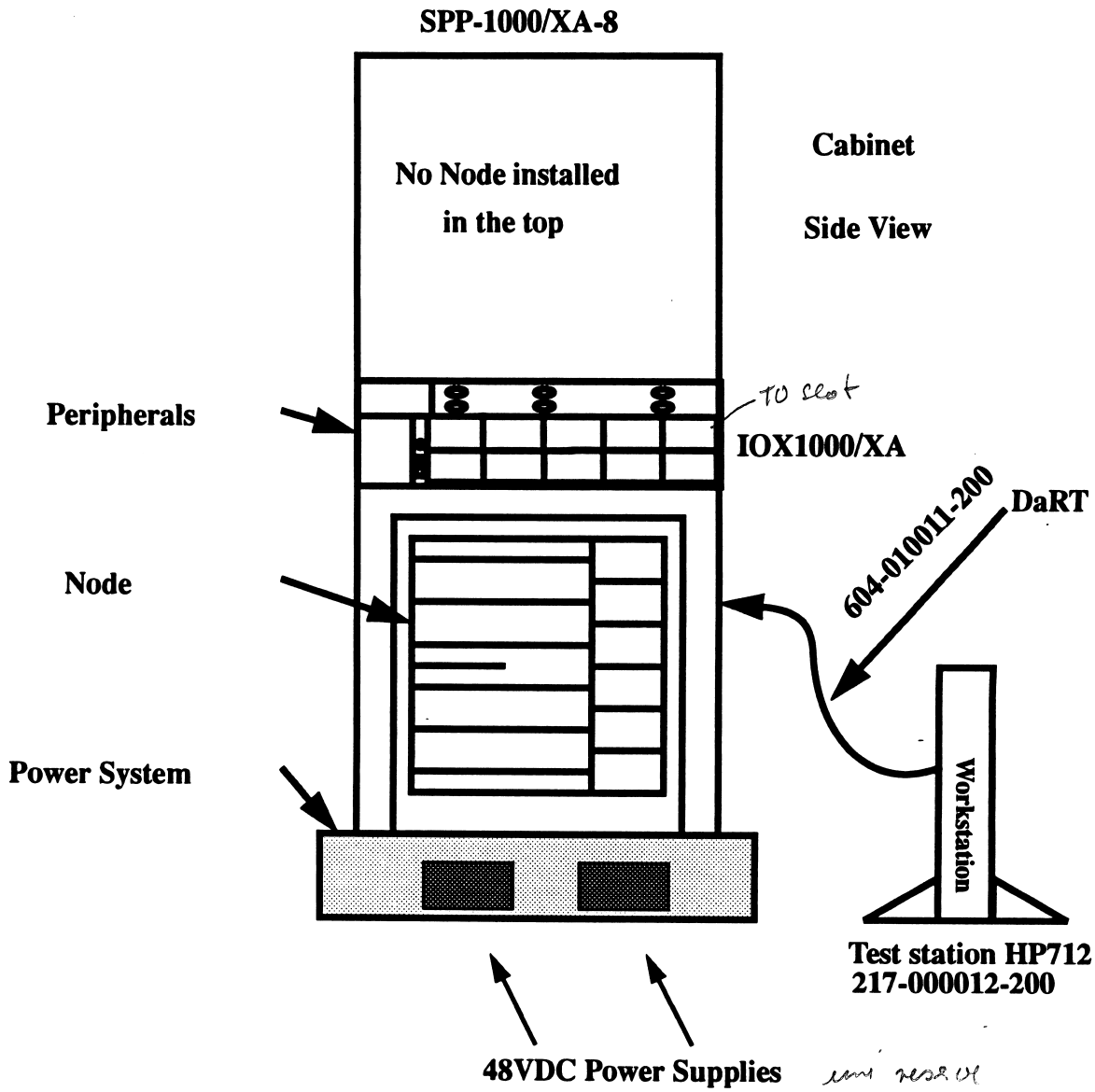
This chapter will cover the following SPP-1000/1200 configurations:

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

2.2 XA Single Node

- XA - eXtended Architecture
- Single node XA weighs approx. 500 lbs

Figure 2-1
XA - Single Node Configuration



2.3 XA 2 Node Configuration

- Dual node XA cabinet weighs approx. 650 lbs.
- Each node has it's own LCD panel.

Figure 2-2
XA - 2 Node Configuration

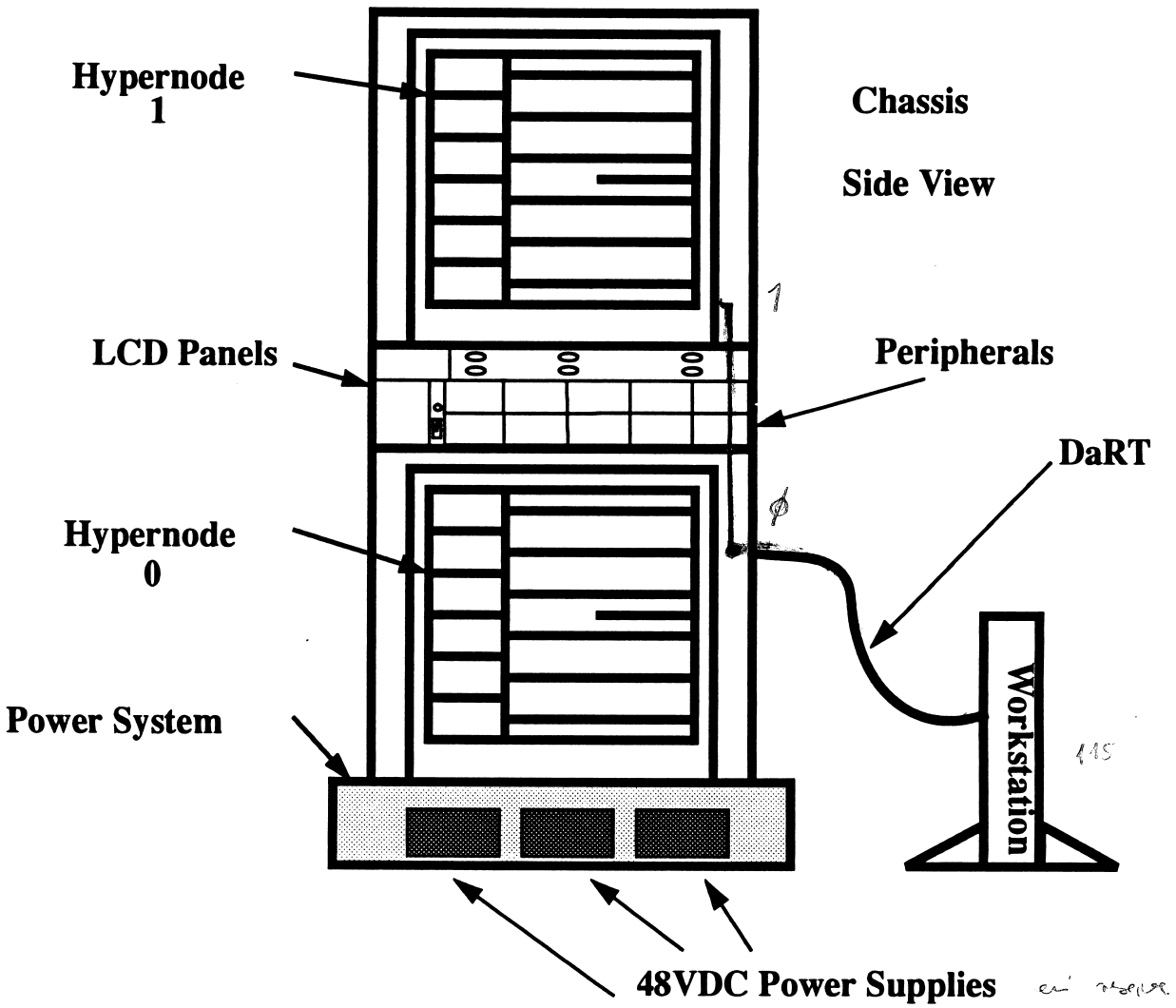
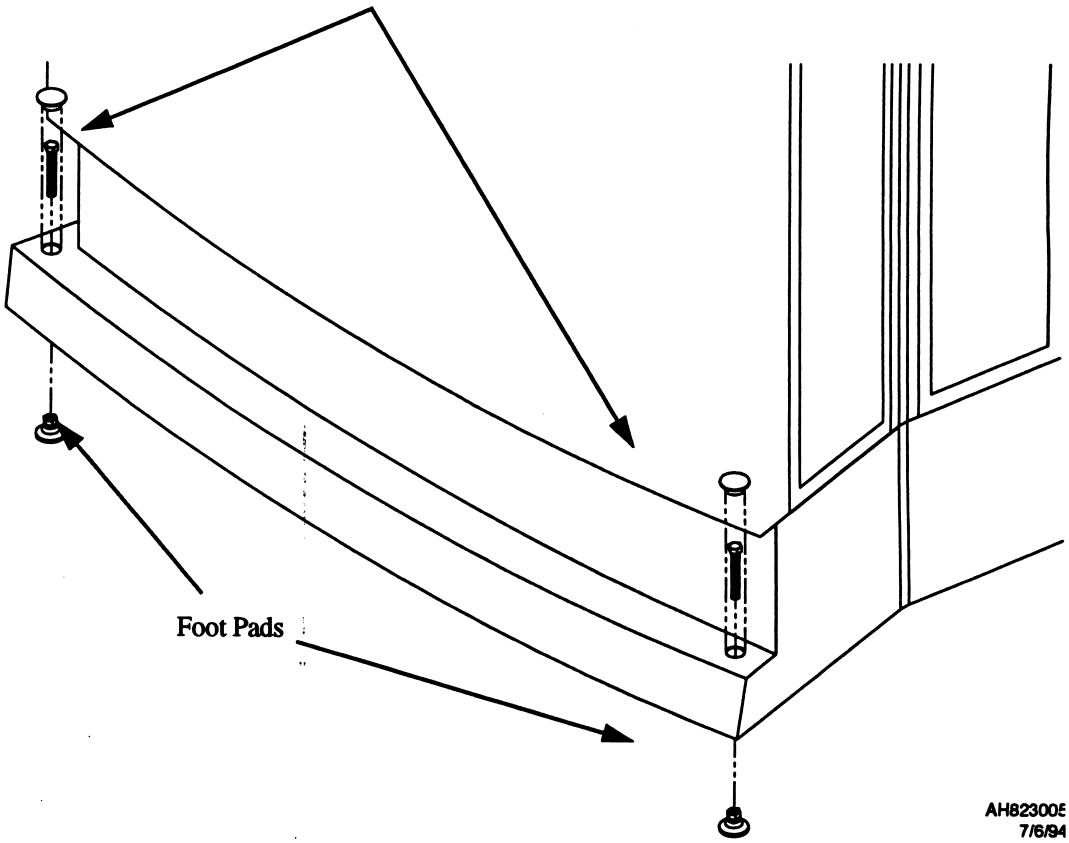


Figure 2-3
XA Leveling Feet

The XA Cabinet leveling feet are installed from the top.



2.4 CD Single Node

Figure 2-4
CD Single Node configuration

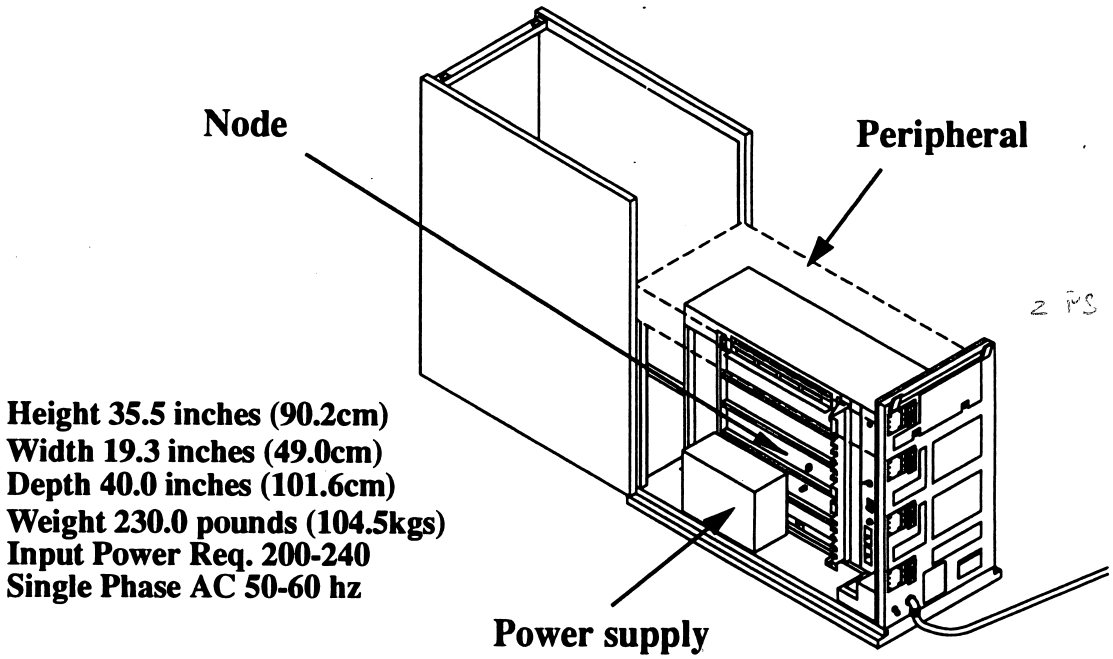
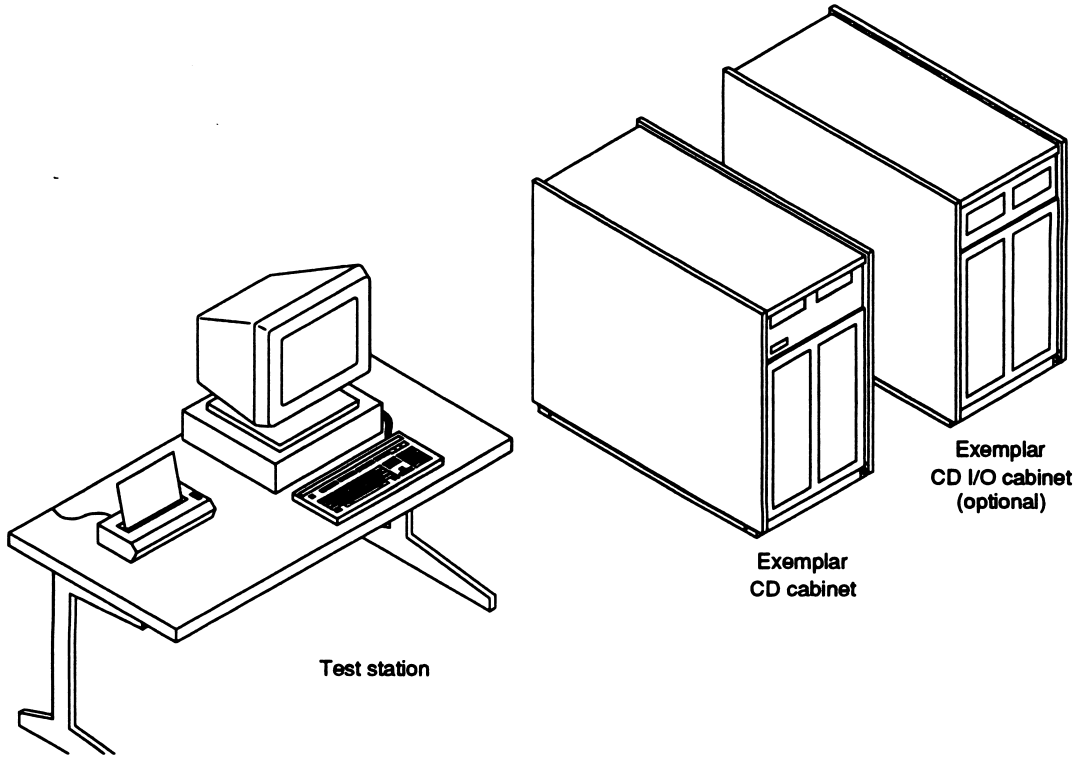


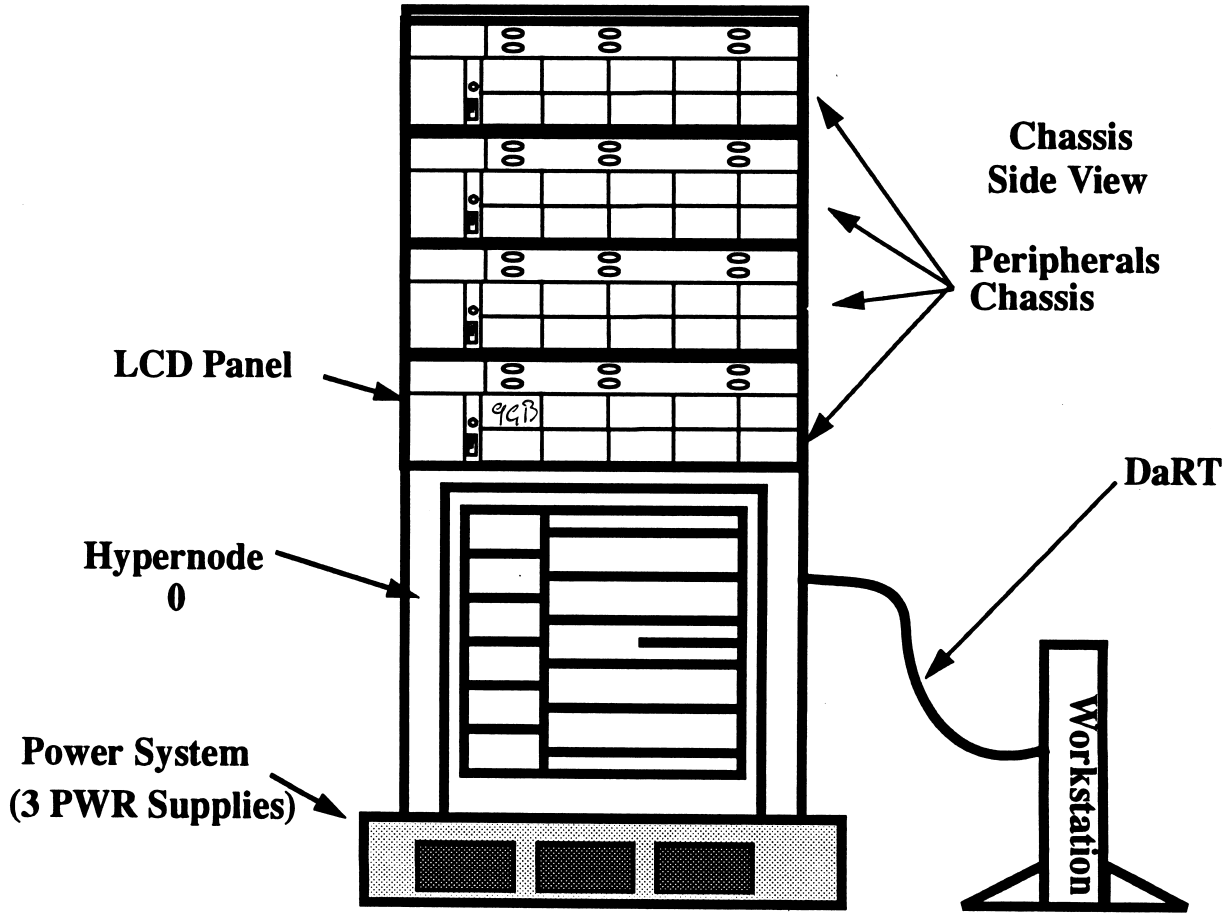
Figure 2-5
CD W/Test Station and IO chassis



2.5 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 2-6
XA Single Node Multi-I/O configuration



2.6 CD Multi-IO

- The I/O Chassis on the CD can be a differential or single ended SCSI.
- The I/O Expansion cabinet MUST contain the High Density (XA type) IO chassis.

Figure 2-7
CD - Multi-IO

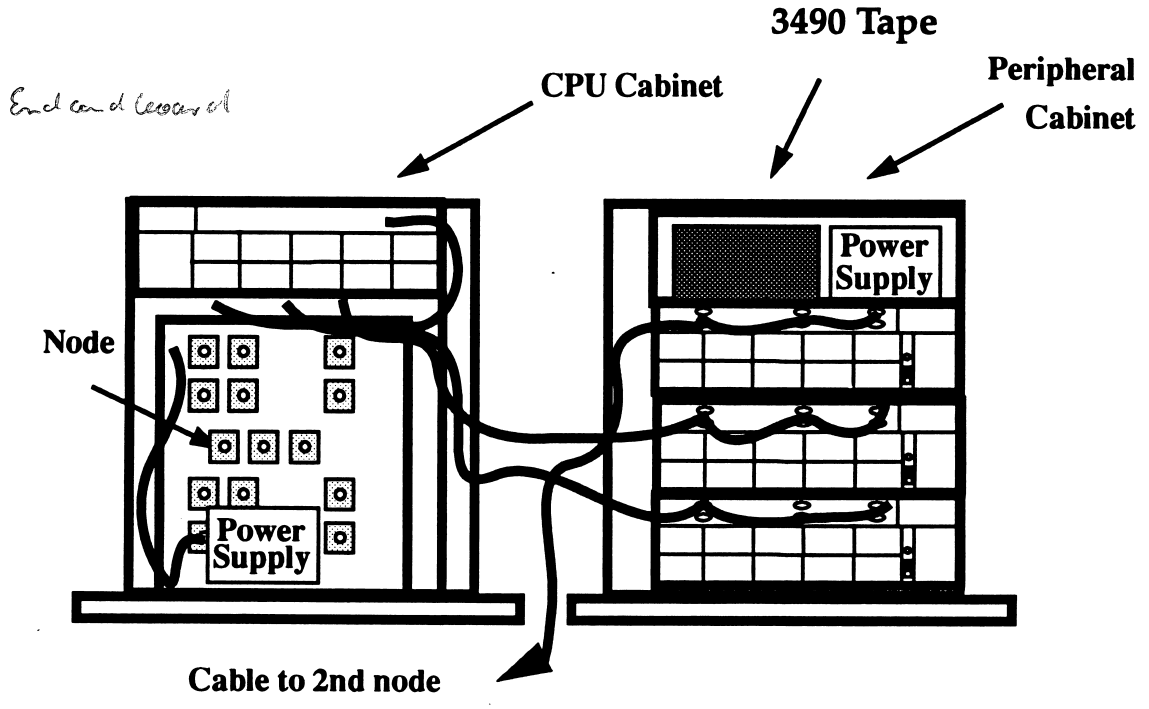
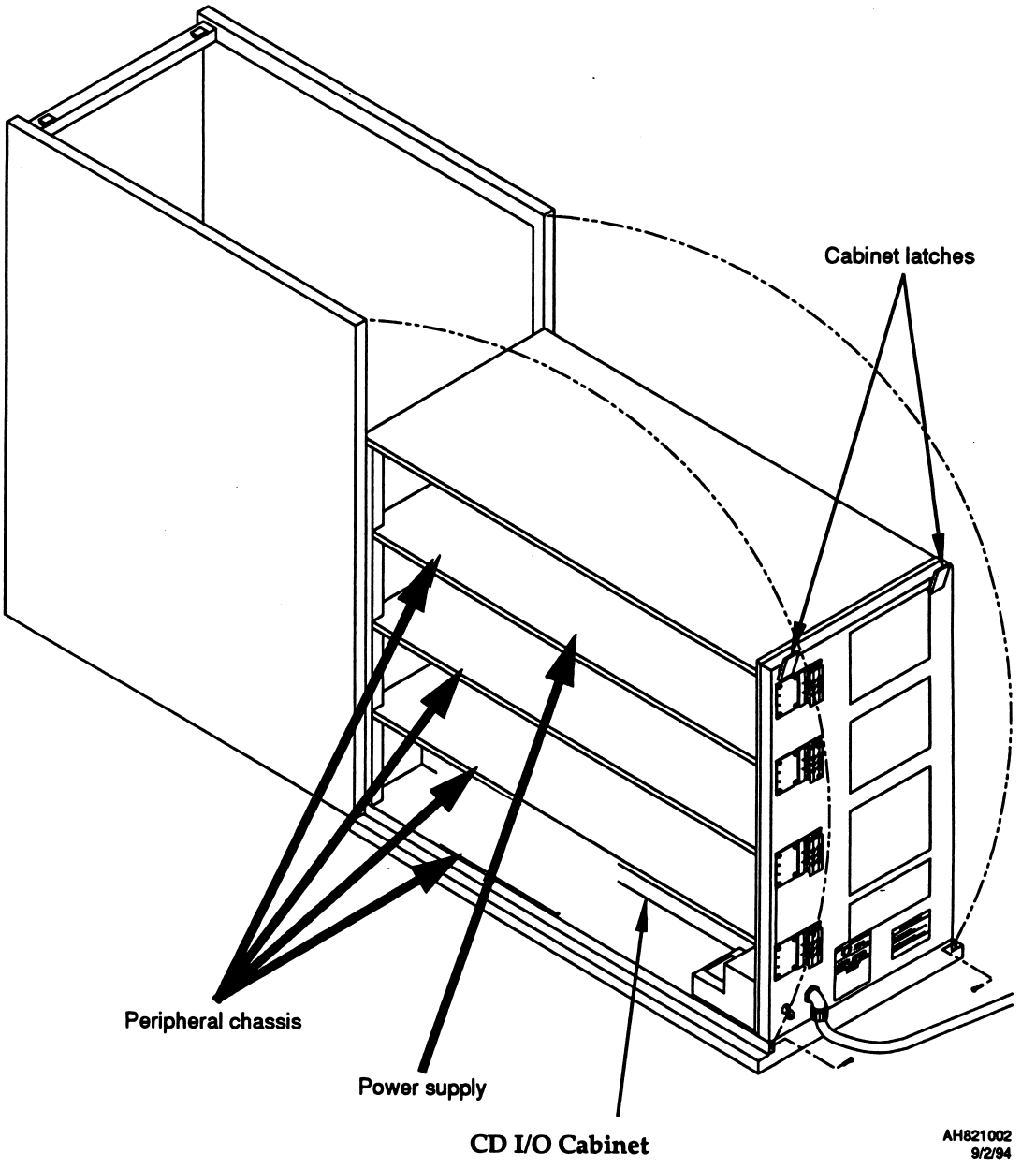


Figure 2-8
CD - I/O Chassis and cover

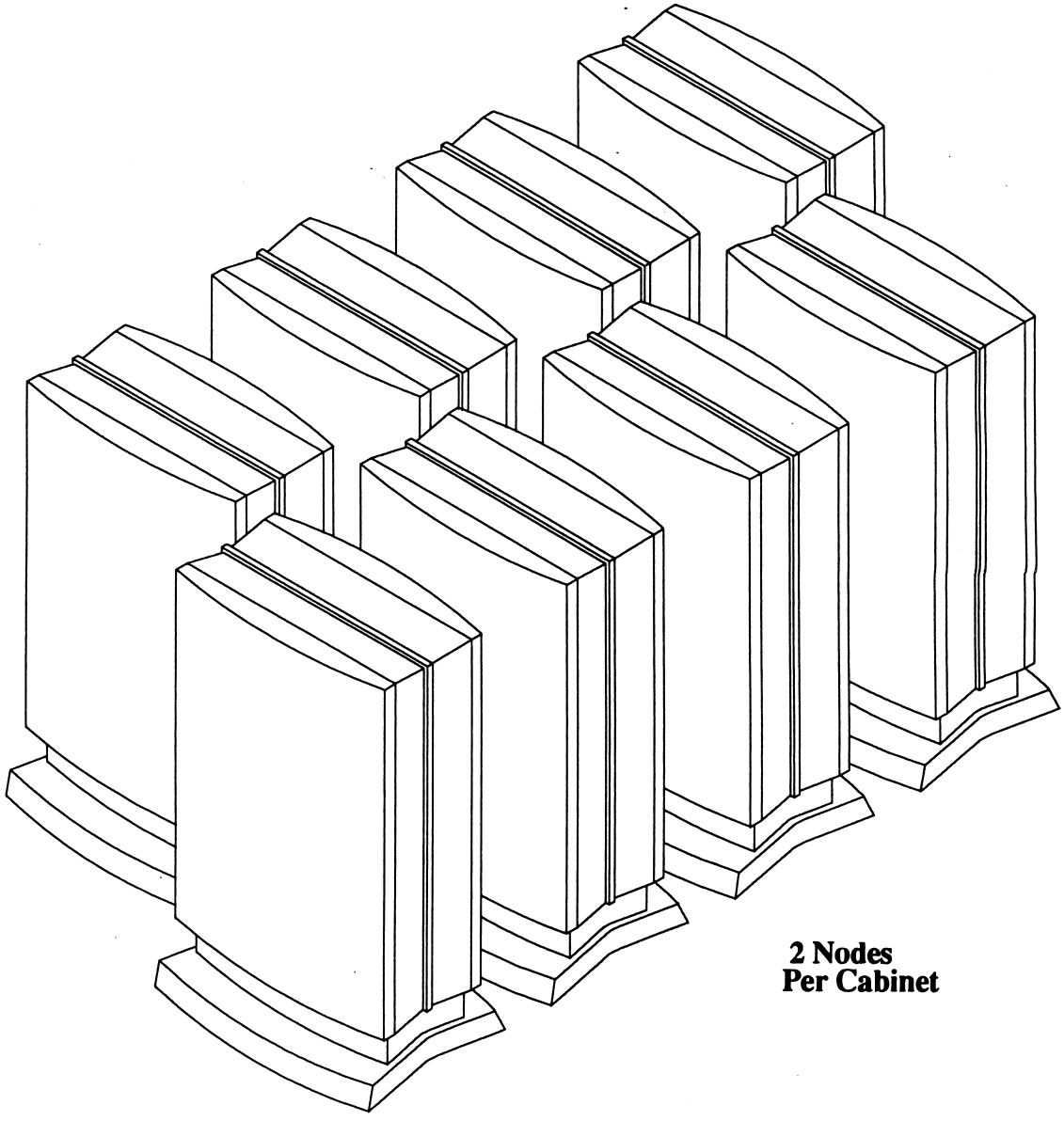


2.7 XA - Max Configuration

- 8 CPUs Per Chassis
- 16 CPUs Per Cabinet
- 128 CPUs Maximum configuration of a Complex (16 Nodes) 16 Nods x 8 CPU = 128 CPUs
- 32 Gigabytes memory w/ 512M MTVs
- 64 Gigabytes w/ 1Gig MTV (N/A Yet) 64 MB Chips

Figure 2-9
16 Node Configuration

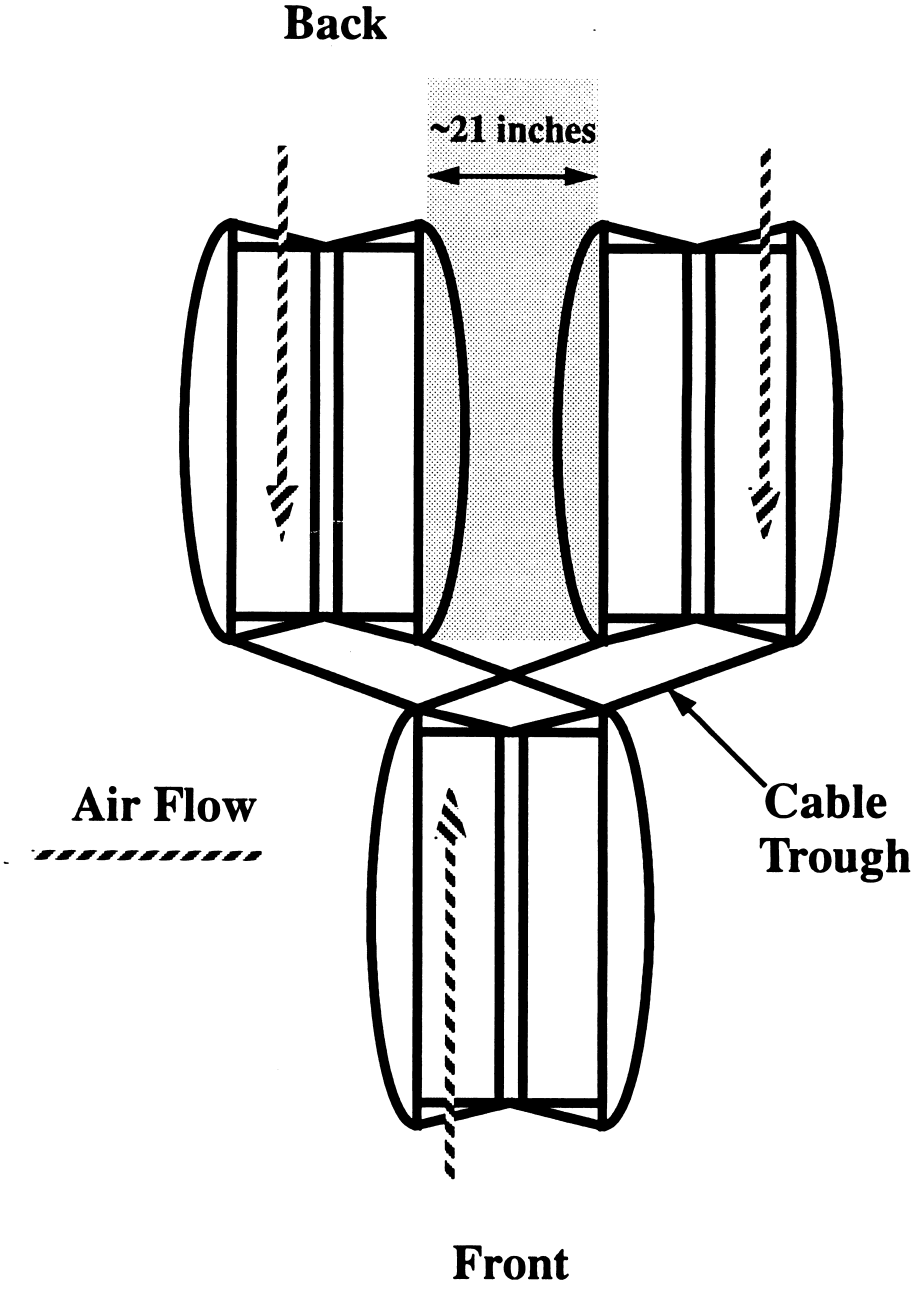
Maximum Configuration



AH500021
10/15/93

2.8 Standard 6 Node Configuration

Figure 2-10
XA 6 Node Configuration



The DaRT Bus 3

3.1 DaRT bus

- The Diagnostic and Remote Test (DaRT) bus is an ethernet like bus that is used by the Test Station to communicate to the nodes.
  - 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, addressing each with a different ethernet address.

Note: The 2.0 diagnostic release established the "OPEN DART" interface. All of the previous releases did not support "OPEN DART". Any MU firmware that is below 2.0 will fail to communicate with a test station running 2.0 or greater.

3.2 DaRT and /etc/host

- The /etc/host 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 3-1
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)**

3.2.1 DaRT Cable

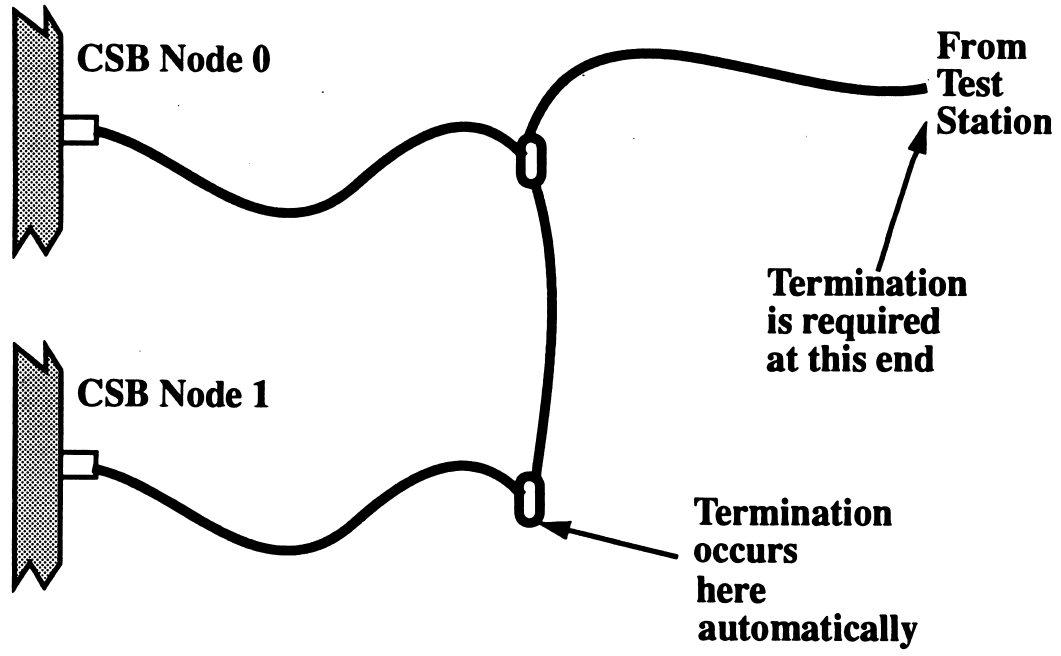
- The standard DaRT bus cable sent with each node in an XA cabinet is a self-terminating cable.
  - There is no need to terminate the end of the cable that is attached to the node.

- It has been seen in the field where the self termination on the 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.

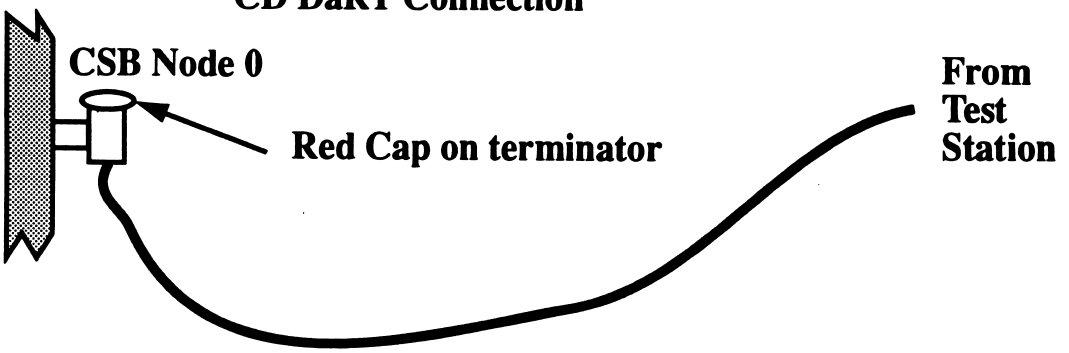
- The following figure shows how the DaRT bus cable is configured

Figure 3-2
The DaRT bus cable

Note: A special cable is shipped with each node. The cable is self terminating allowing easy multi-node connections.



CD DaRT Connection



Ethernet address board has been removed from USA. The Ethernet address must be installed

Front Panel LCD 4

4.1 Front Panel Display

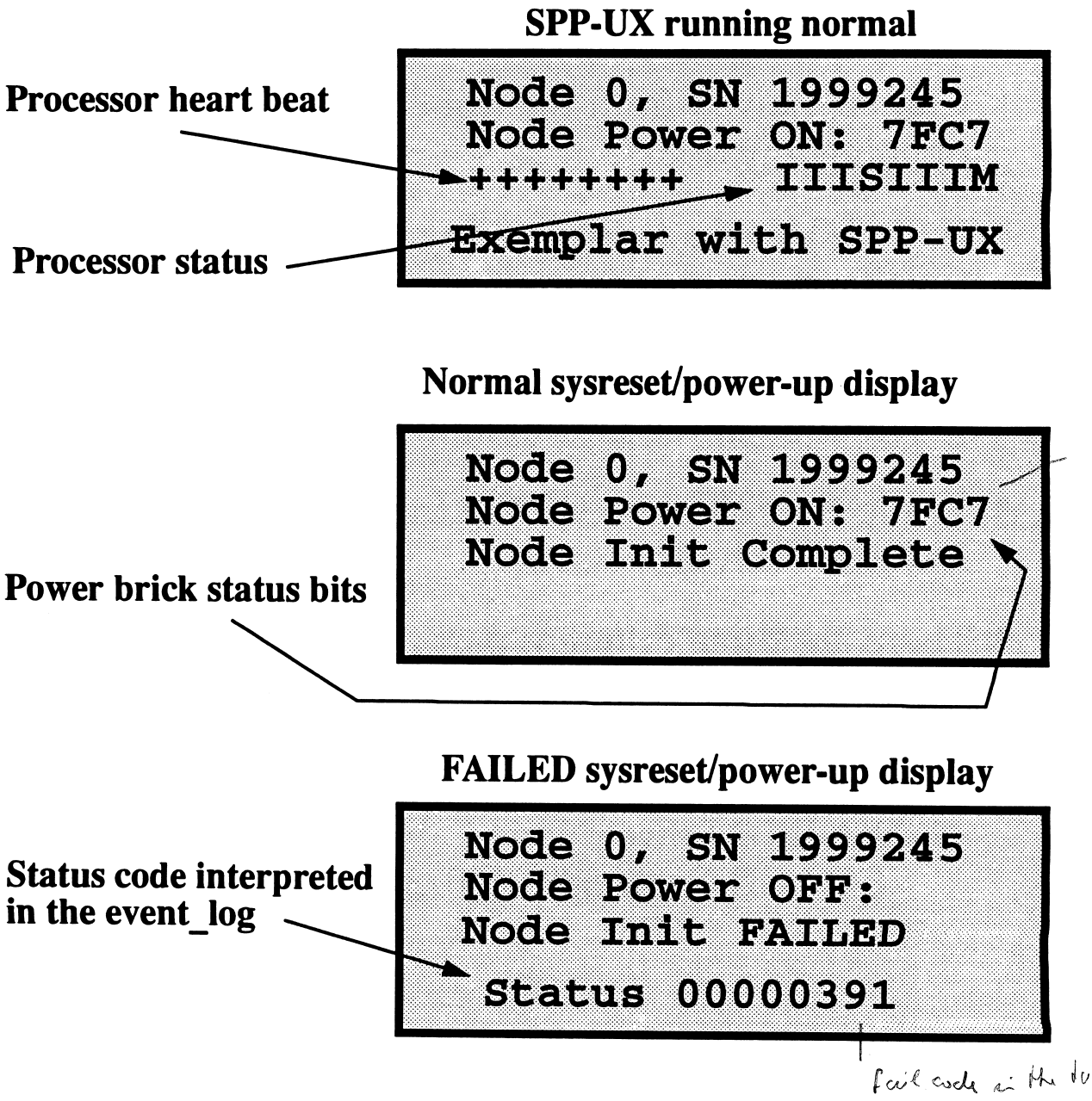
Each node has a Liquid Crystal Display which is 4 line deep and 20 characters wide.

- 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 is 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 what 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

4.2 Typical Front Panel Messages

- In the following figure the front panel messages are examples of some types of messages that can be seen after events such as:
  - power on, sysreset, MU selftest failures or booting SPP-UX.

Figure 4-1
Front Panel Messages



CSB/ASB Level Functional Units 5

5.1 Introduction

This chapter will cover in more detail each functional block contained within the CSB/ASB that pertains to the gate arrays attached to the system board.

5.1.1 CSB4 Physical Layout

- The CSB has both logic boards and gatearrays directly attached it.
- Figure 5-1 on page 32 and Figure 5-2 on page 33 show the physical layout of the CSB
  - Part number 410-001396-200.

Camelot

Figure 5-1
Camelot ——→ CSB Layout Z-Side

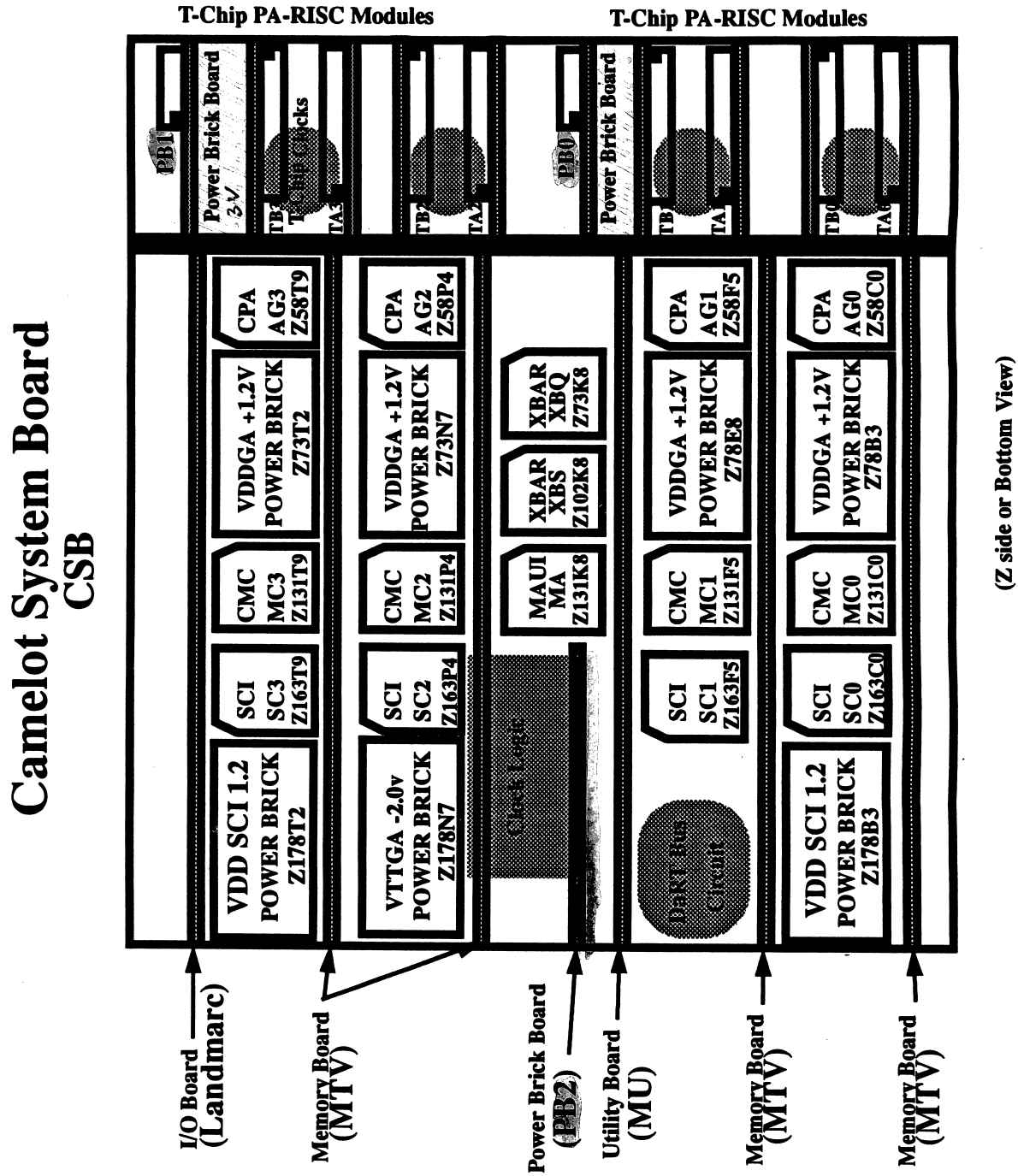


Figure 5-2
CSB U-Side Layout

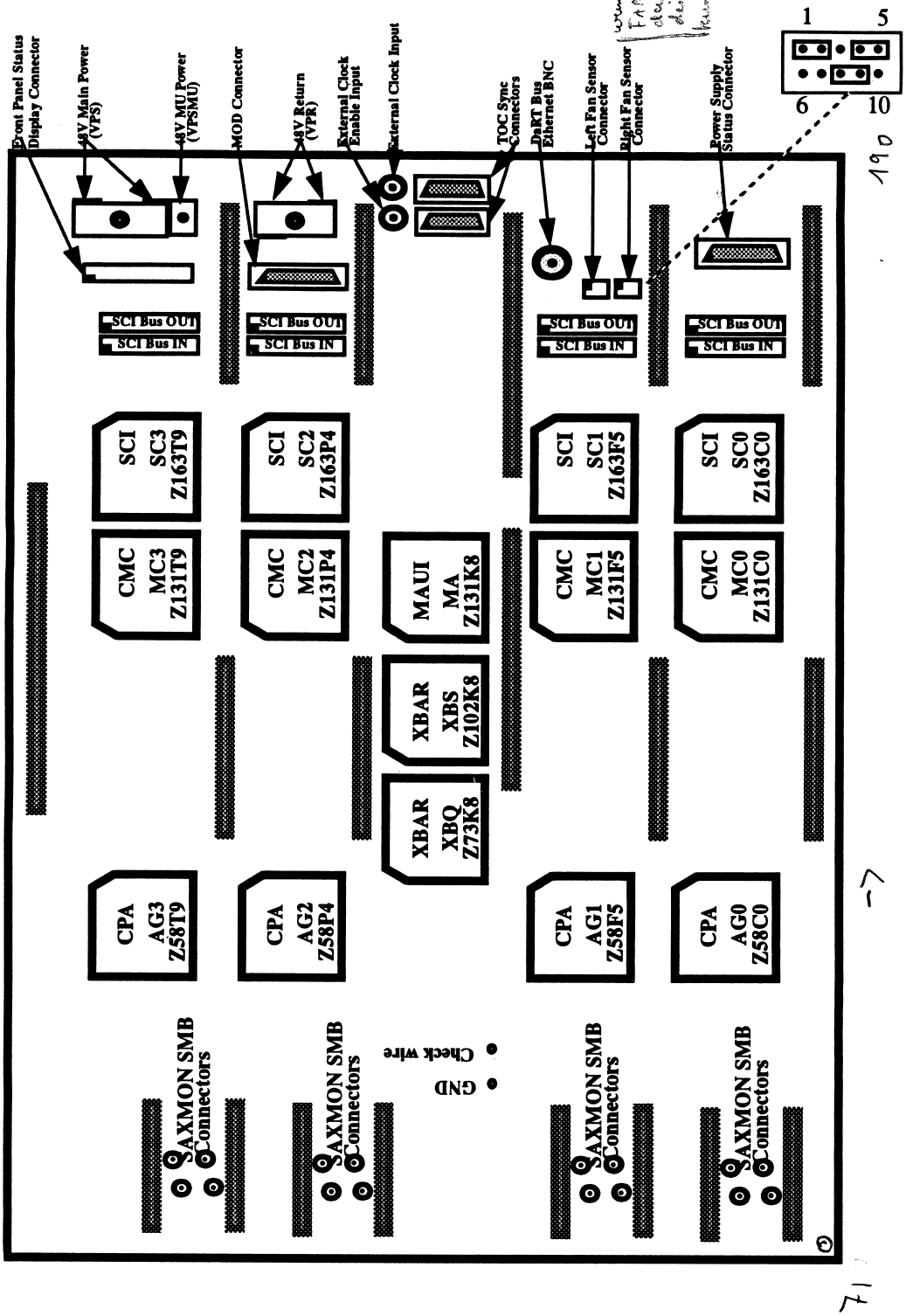
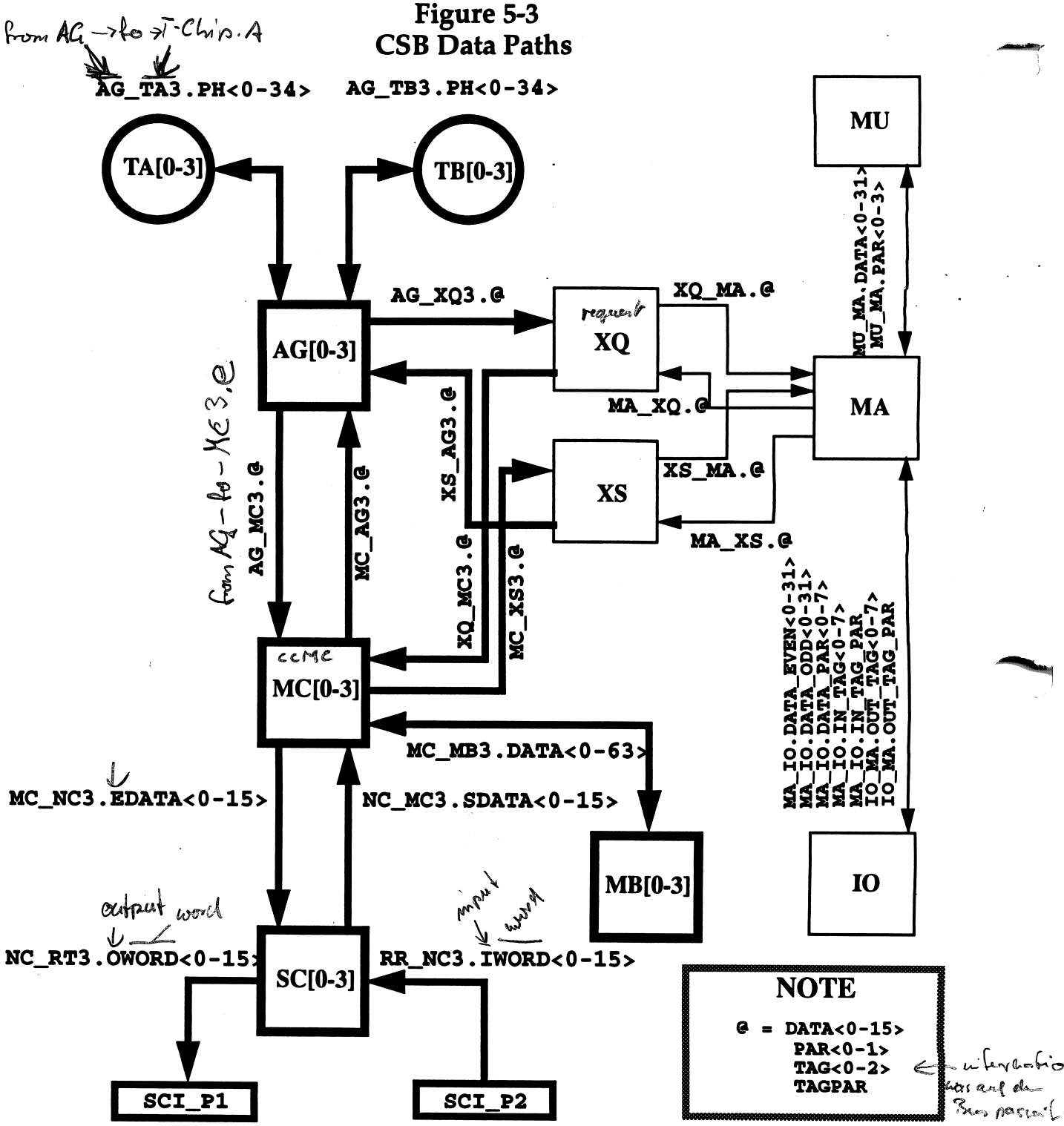


Figure 5-3
CSB Data Paths

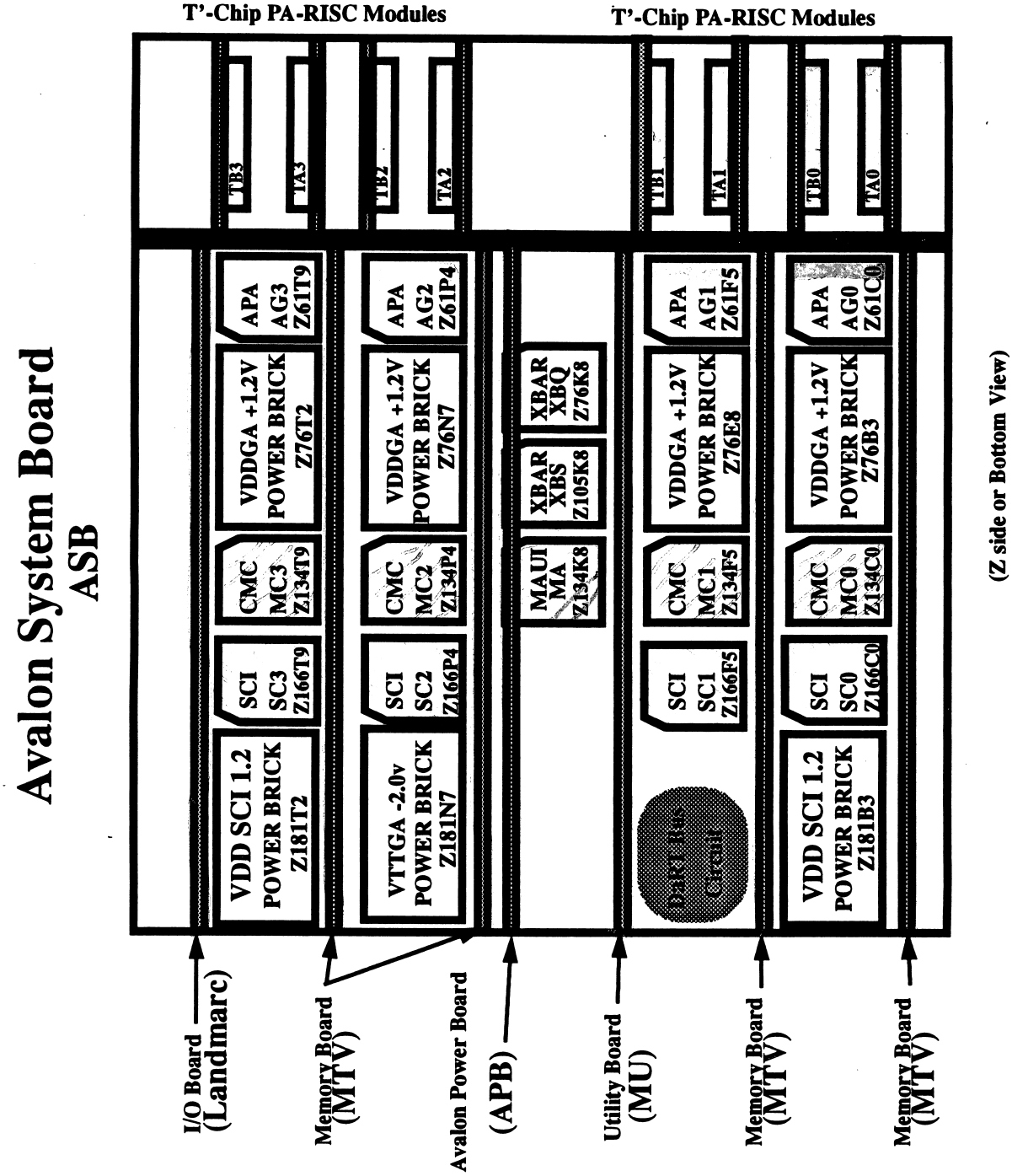


Avalon

5.2 ASB Physical Layout

Avalon

Figure 5-4
ASB Physical Layout



- Must have APAs and XBAR2 GAs

Figure 5-5
ASB U-SIDE

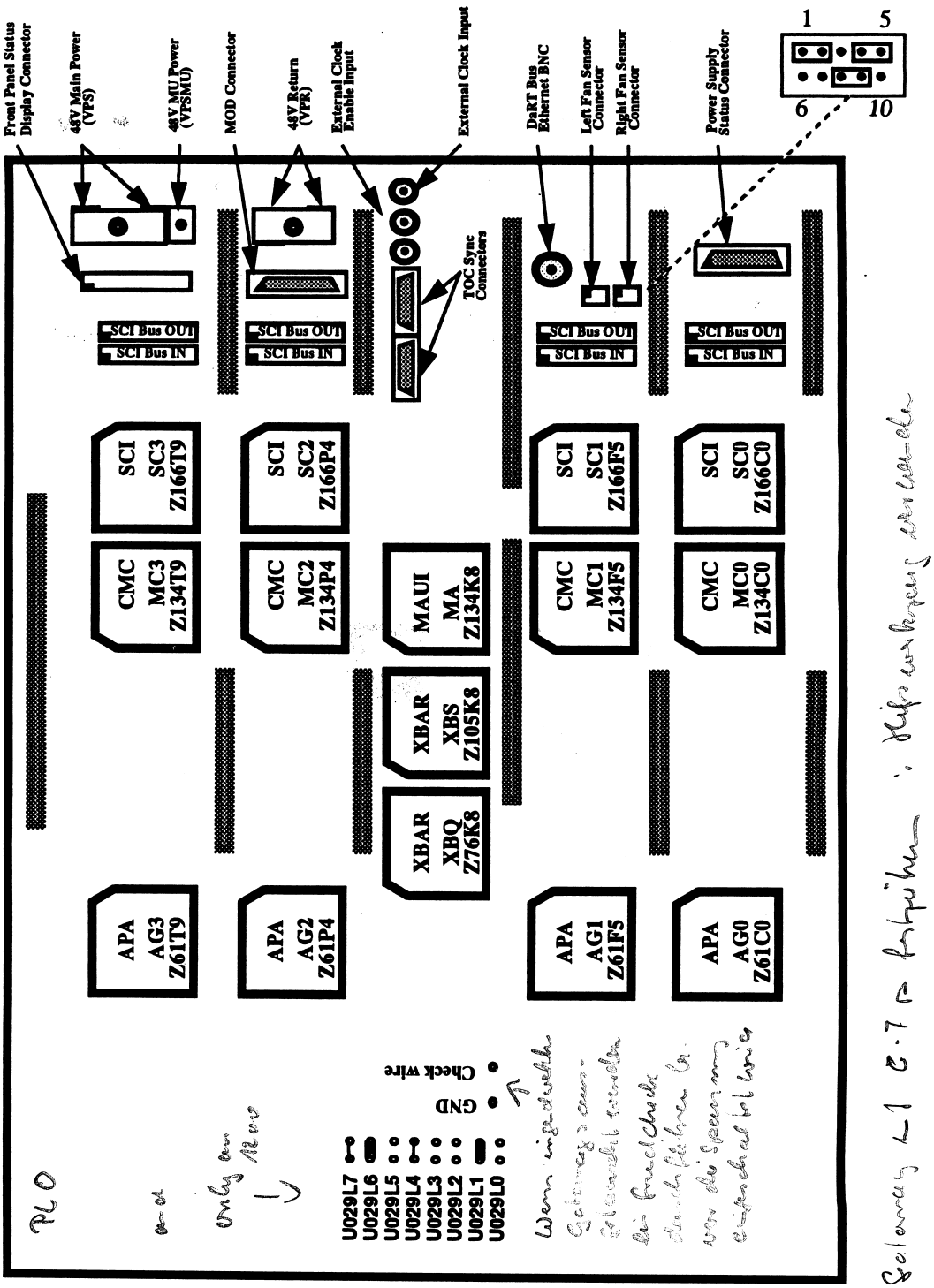
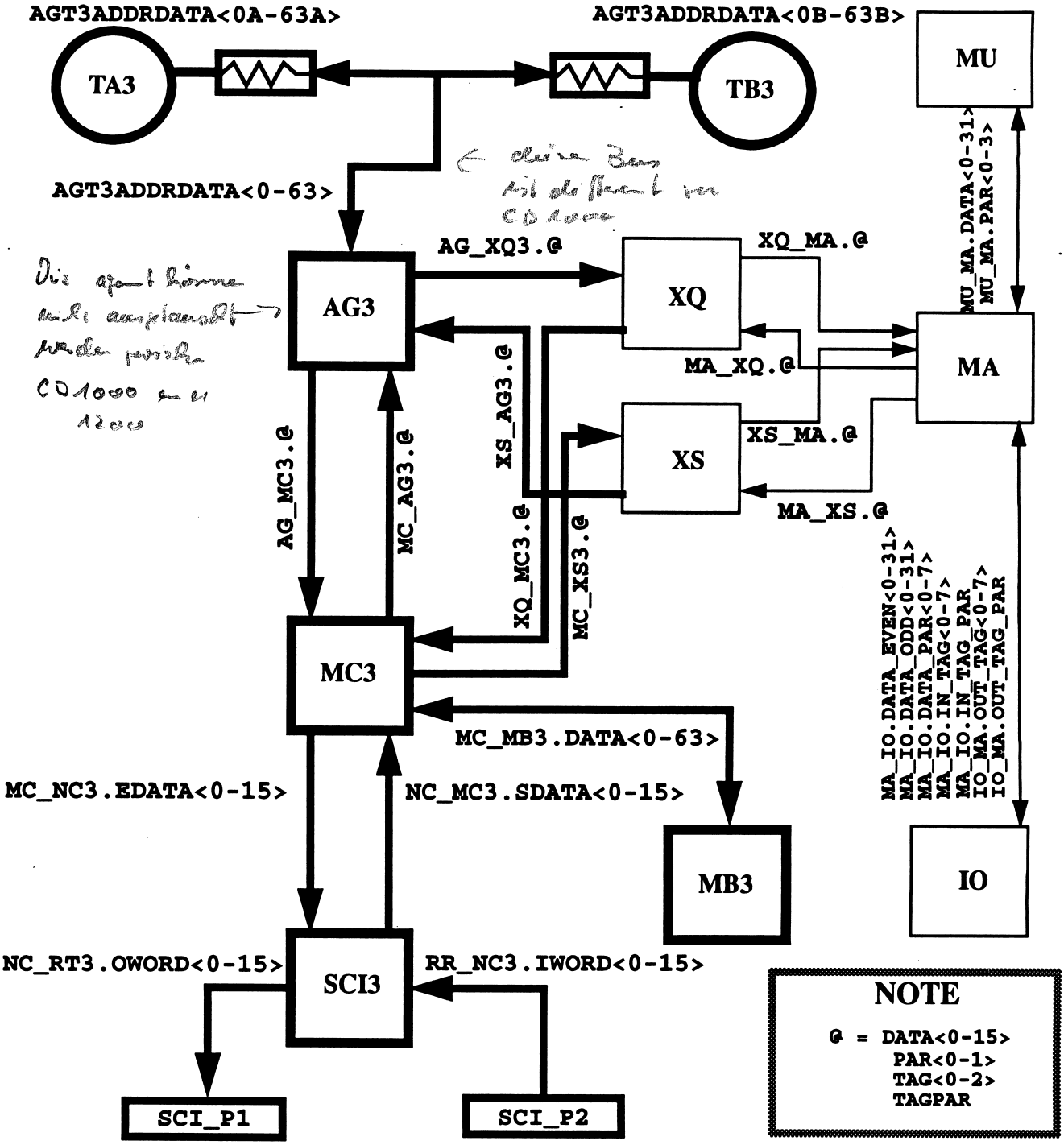


Figure 5-6
ASB Data Paths



** Signal names use slice 3*

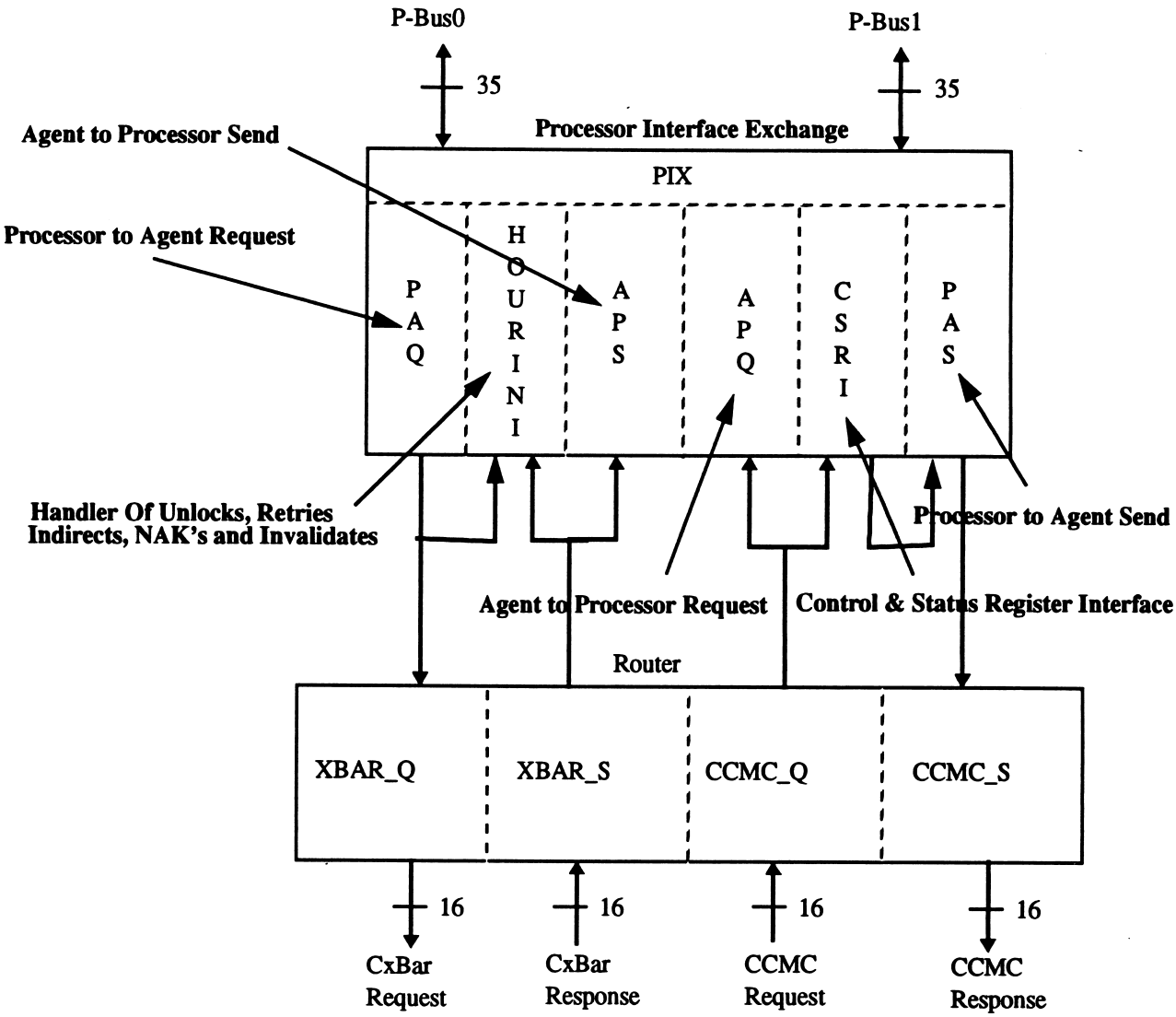
5.3 Agent Gate Array

The Agent gatearray is the interface between the processors in a slice and the rest of the system.

- It communicates directly with the memory control gatearray and the crossbar memory arrays.
- The silk screen on the CSB designates it as (AG).
- There is one Agent per-slice or 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 39 shows a block diagram of the Camelot Processor Agent gatearray (CPA).

Figure 5-7
Agent Gate Array Block Diagram

APA



****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.*

5.3.1 Agent Hard Errors

Agent:

Table 5-1 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 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 Imiss
cpa/paq_mts_err	PAQ Message Transfer Sequence Error
cpa/paq_sbdt_adr_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

Table 5-2 External errors

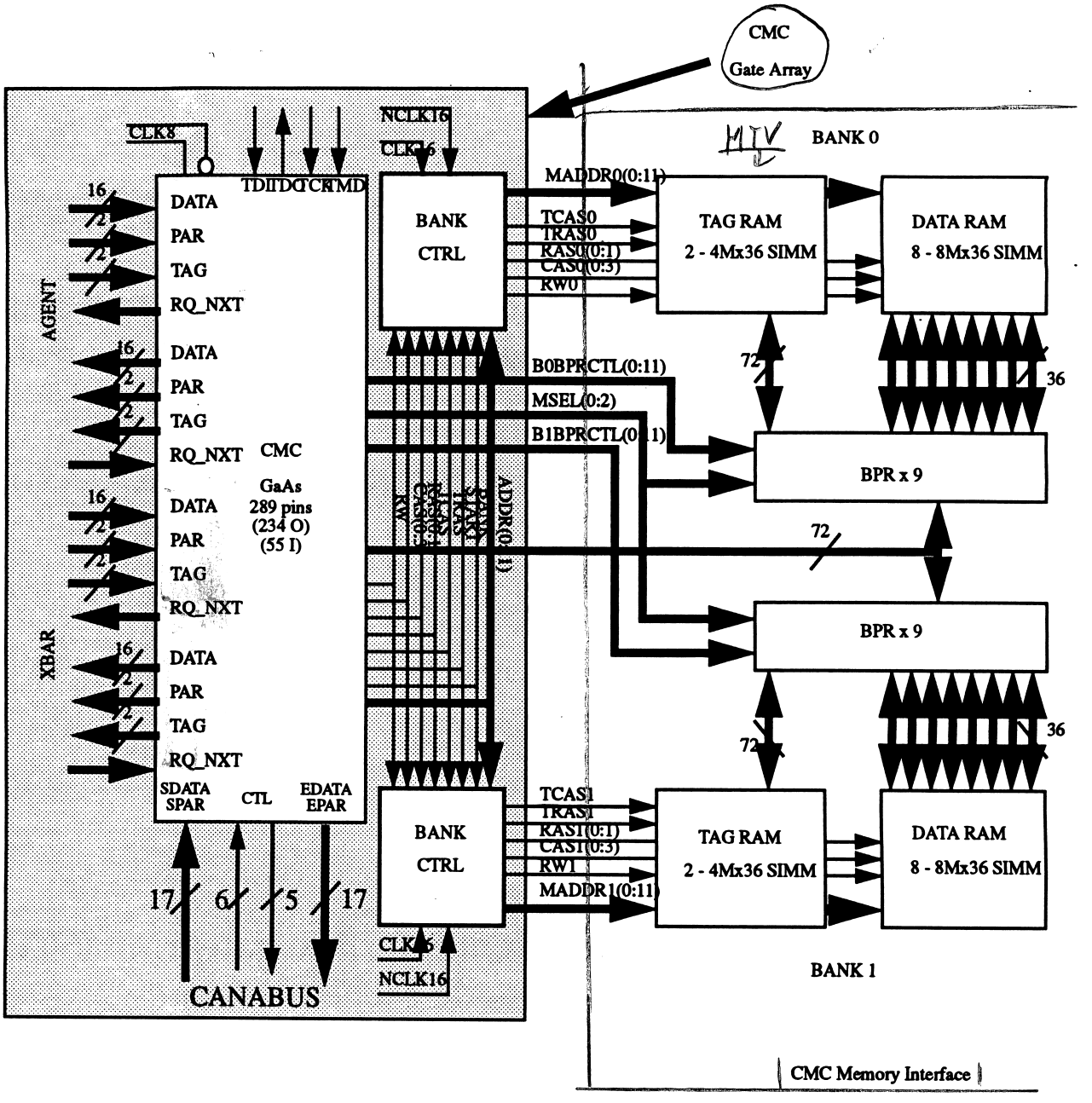
cpa/hpmch_aps_dque_rd_par_err	HIGH PRIORITY MACHINE CHECK aps_dque_rd_par_err
cpa/q_tag_seq_err	Tag sequence error on CMC request port
cpa/s_data_par_err	Byte parity error on XBAR response port

5.4 CCMC Gate Array

The Convex Coherent Memory Control gatearray 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 gatearray (SCI)
    - which enables node cache coherency.
- Figure 5-8 on page 42 and Figure 5-9 on page 43 are block diagrams of the CCMC.

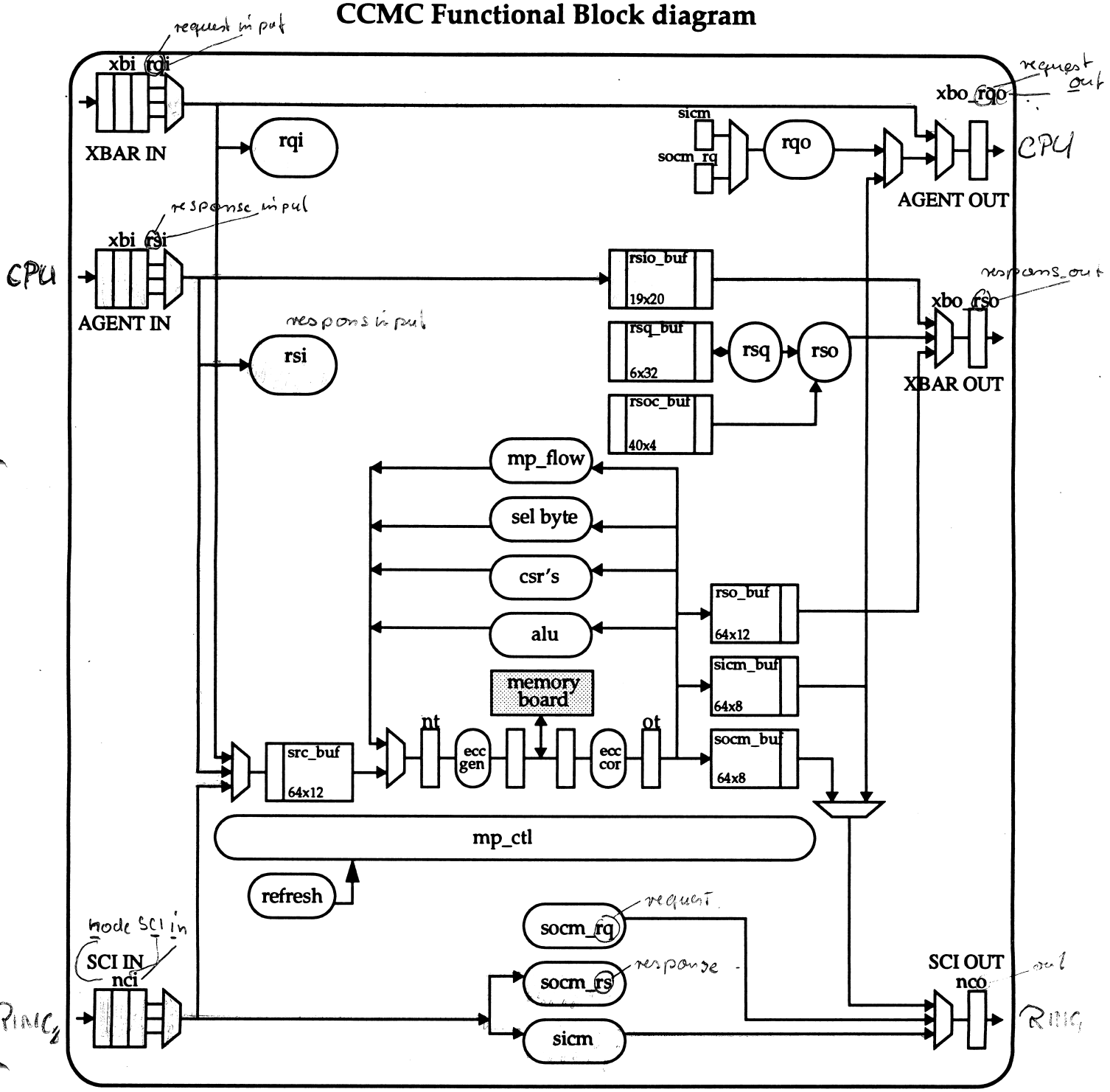
Figure 5-8
CCMC Memory Interface Block diagram



5.4.1 CCMC Functional Block diagram

The following figure is a more detailed block diagram of the CCMC.

Figure 5-9
CCMC Functional Block diagram



5.4.2 CCMC Hard Errors

Table 5-3 CCMC Hard Errors

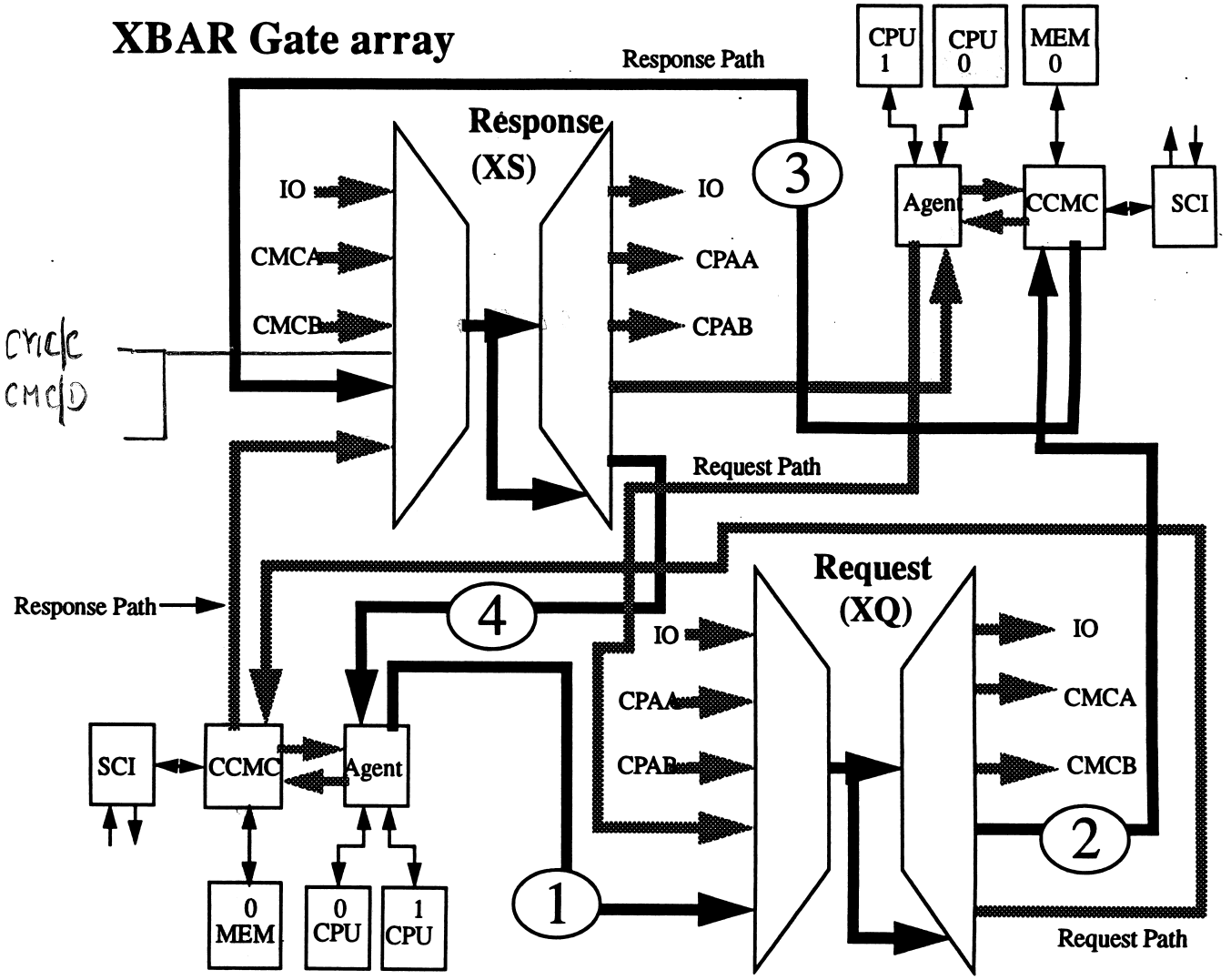
Error Name	Description
RQI_PAR_ERR	Xbar request in, Parity error
RQI_SEQ_ERR	Xbar request in, Bad head-body-tail sequence
RQI_FMT_ERR	Xbar request in, Bad packet length
RSI_PAR_ERR	Xbar response in, Parity error
RSI_SEQ_ERR	Xbar response in, Bad head-body-tail sequence
RSI_FMT_ERR	Xbar response in, Bad 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

5.4.3 Xbar Gate Arrays

The Crossbar function of the SPP system is implemented in two identical Xbar gatearrays.

- **XBQ** - the **XB**ar re**Q**uest handles all of the Requests.
- **XBS** - the **XB**ar re**S**ponse 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.

Figure 5-10
Xbar Block Diagram



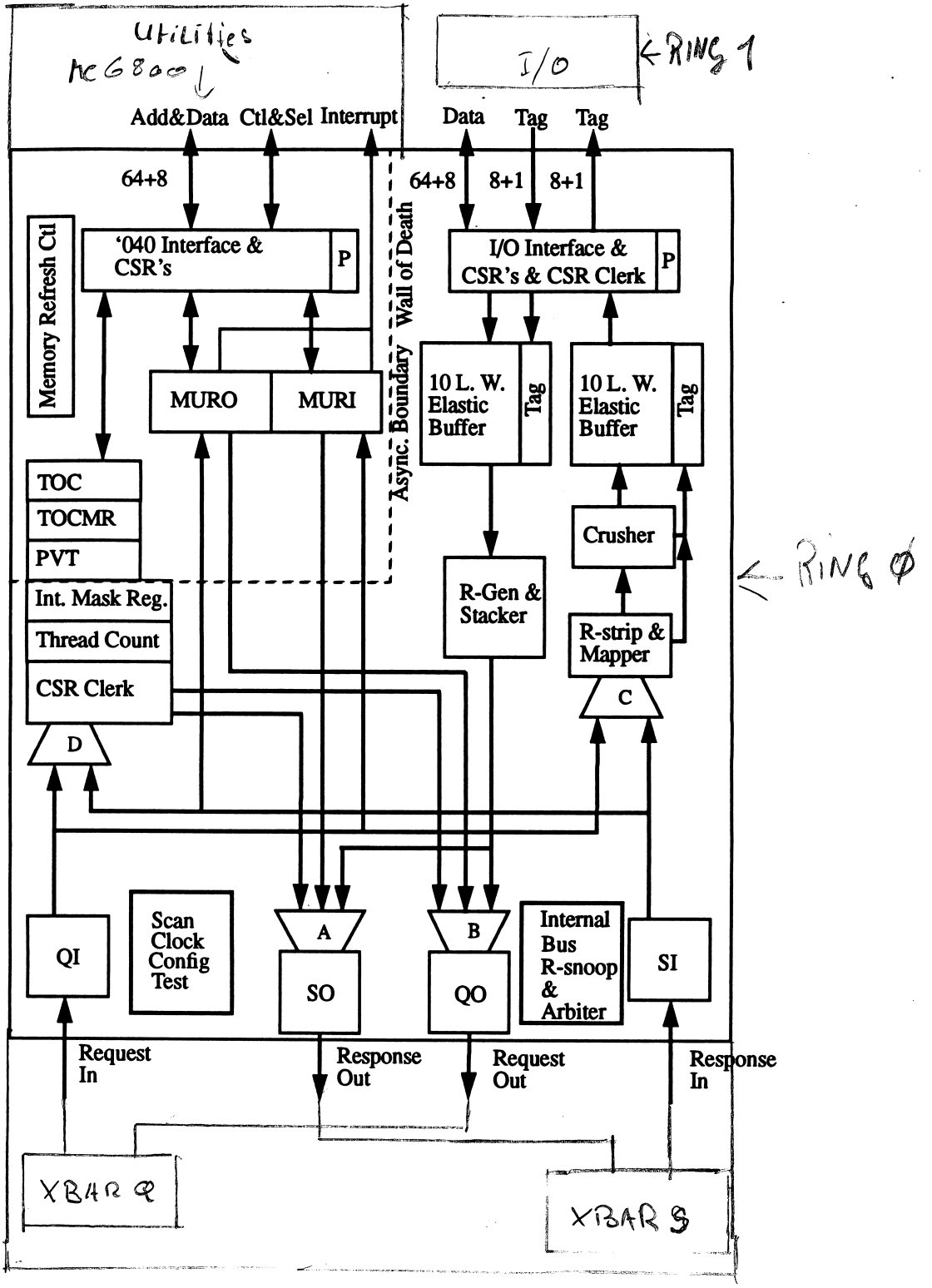
1) Agent (Port C) sends a Request to the Request Xbar (XQ).
2) The XQ routes the Request to the target CCMU (Port C).
3) The Target CCMU sends a Response to the Response Xbar (XS).
4) The Response Xbar routes the Response to the requesting Agent.

5.5 MAUI Gate Array

The **M**pp **A**gent **U**tility and **I**/O gatearray 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 gatearray also contains:
  - the System Control and Status Register
  - Time of Century Counter
  - Thread Execution timers
  - Memory refresh logic.
- Figure 5-11 on page 48 is a block diagram of the MAUI.

Figure 5-11
MAUI Block Diagram



5.5.1 MAUI Hard Errors

Maui:

Table 5-4 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 External Errors

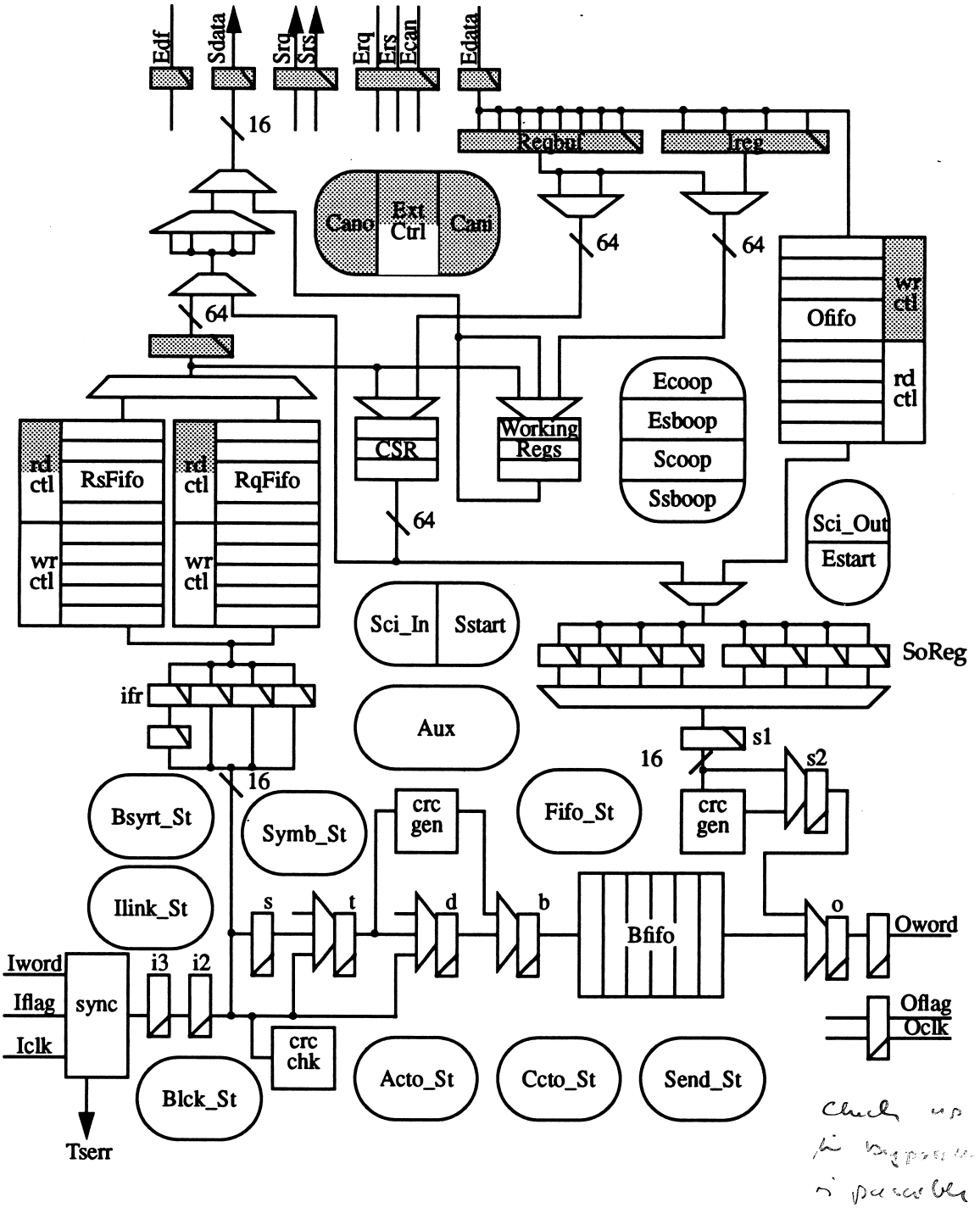
maui/cere_parity_err	to/from Landmarc
maui/rqi_parity_err	From the request xbar
maui/rqi_route_err	From the request xbar
maui/rqi_seq_err	From the request xbar
maui/rsi_parity_err	From the send xbar
maui/rsi_route_err	From the send xbar
maui/rsi_seq_err	From the send xbar

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 memory systems on sperate nodes.

- The SCI arrays will connect the CCMC arrays to the SCI rings.
  - 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.

Figure 5-12
SCI Block Diagram



Note: SCI3s only check the CRC on the target SCI.

Boards 6

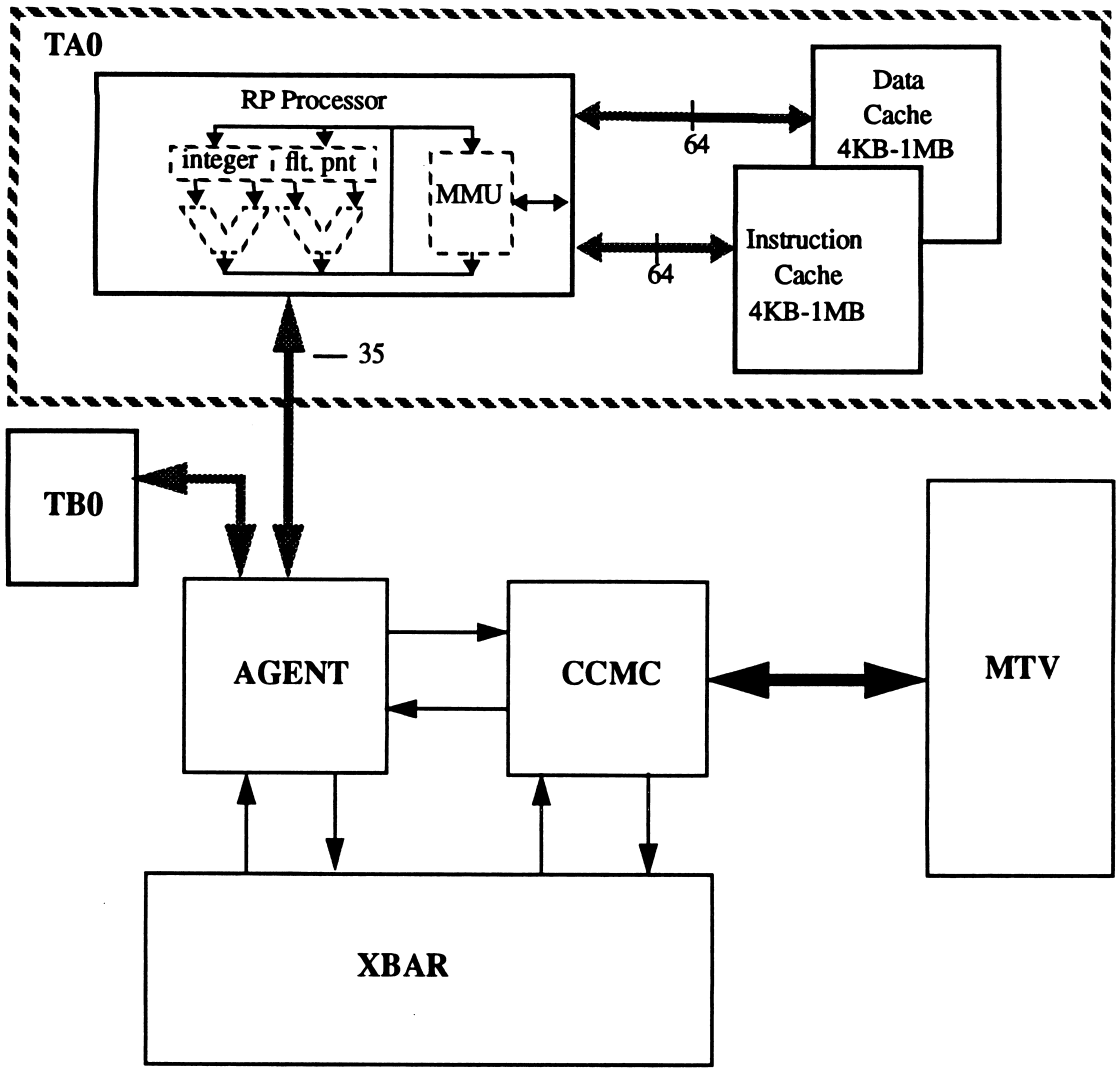
6.1 T-Chip Module - SPP1000

- The T-Chip Module is the processor board that is purchased from HP. It contains:
  - the PA-RISC Processor
  - 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.
- Figure 6-1 on page 55 is a block diagram of the T-Chip Module.

6.2 T' (T Prime) Chip Module

- Used in the spp12000.
  - Contains a 256K Data Cache and ICache.
  - Runs at 120 MHZ .
    - (SPP1200A runs at 100 MHZ)
  - Connects to the APA via a runway bus, which is a true 64 bit bus.

Figure 6-1
T-Chip Module Block Diagram



6.3 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 6-2 on page 56 is a block diagram of the MU.

Figure 6-2
MU Block Diagram

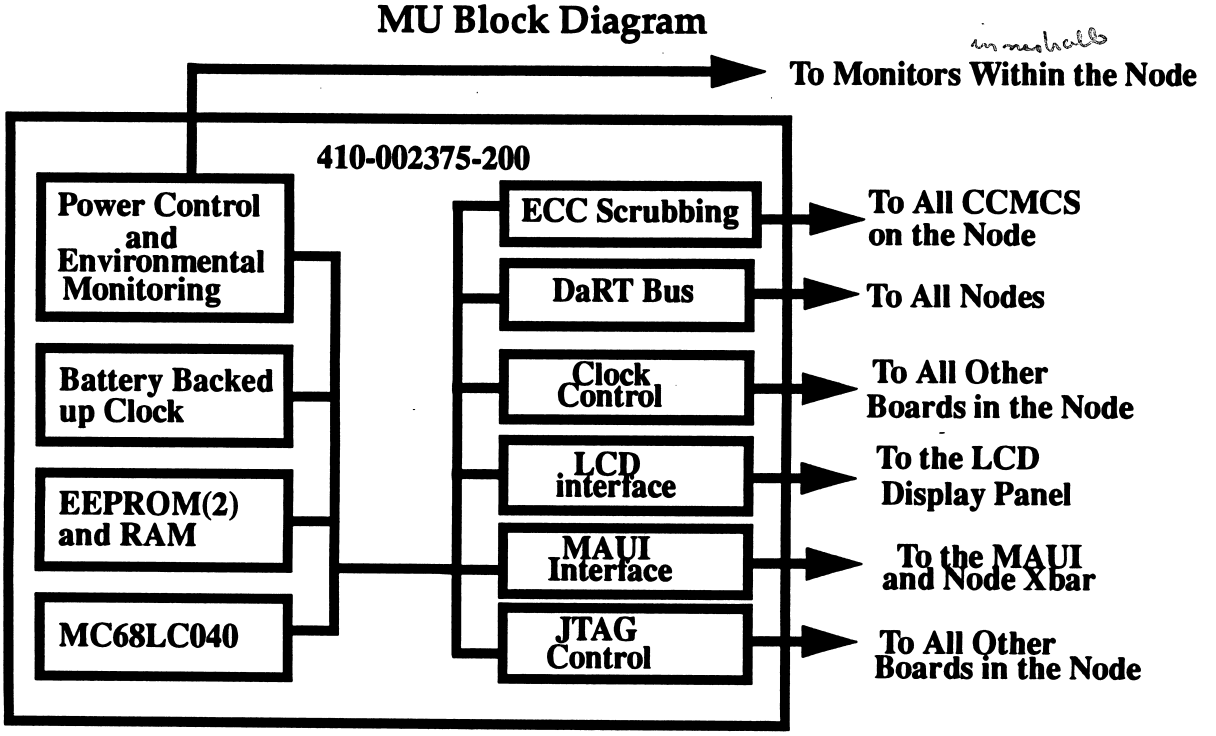
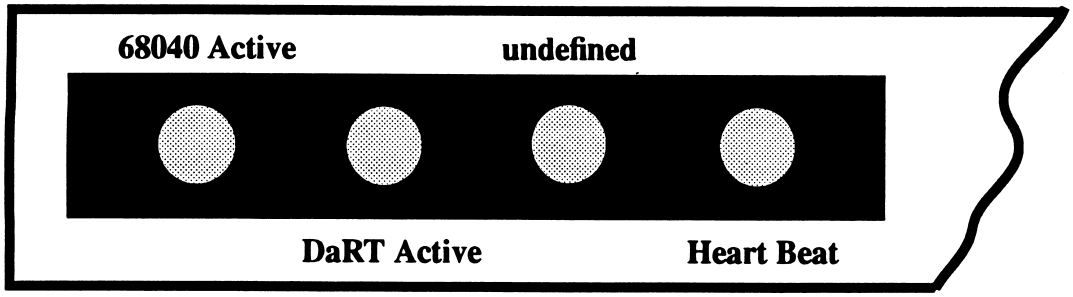


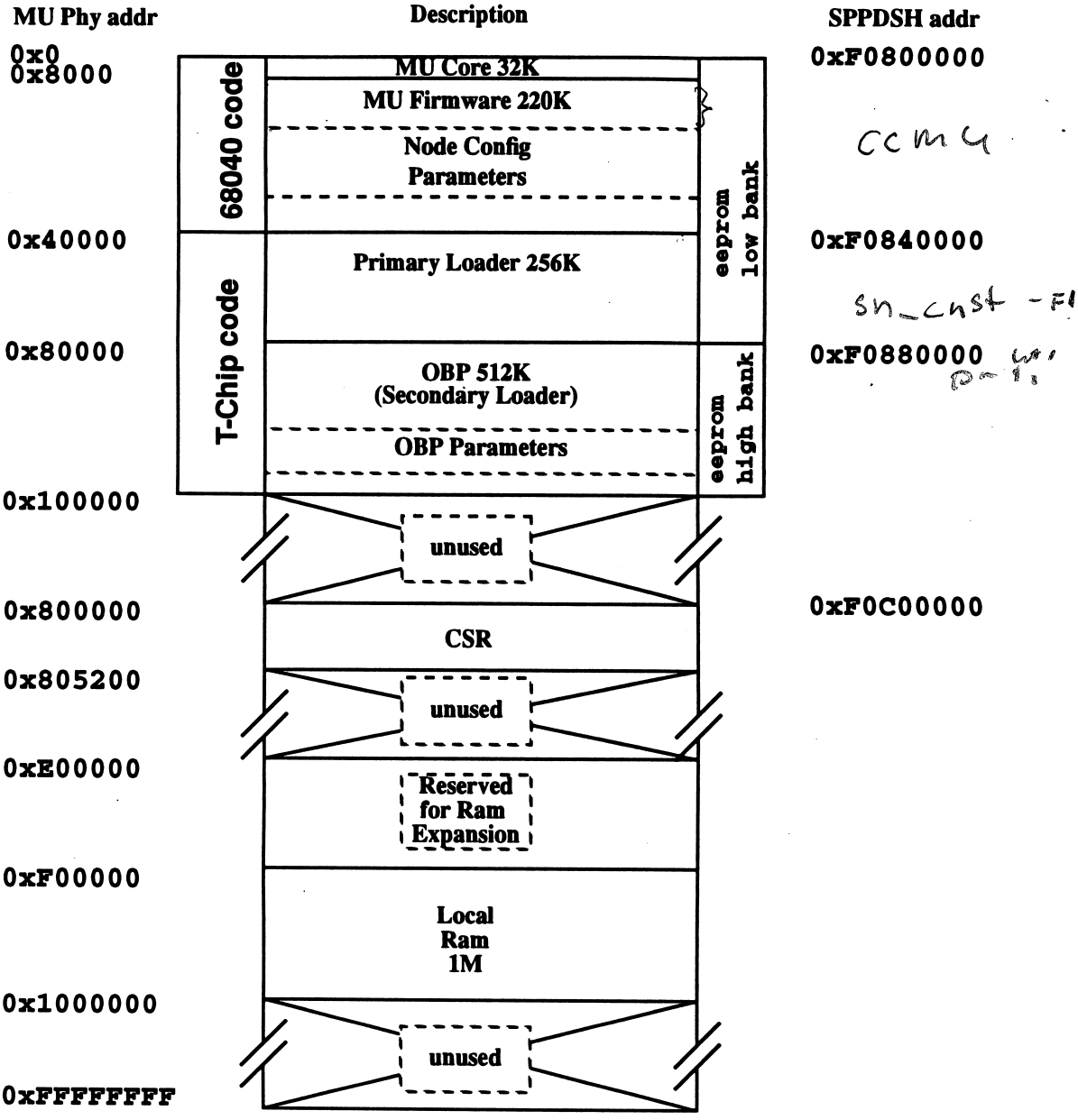
Figure 6-3
MU LEDs



6.3.1 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.
- There are fuses on the MU that should checked on the MU if the MU LEDs do not come on.

Figure 6-4
MU Memory Allocation Map



6.4 Memory Tag Vortex Logic Card

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, therefore 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).
  - The Data Rams contain program data, like data rams in any computer system.
  - The Tag Rams contain important information about the data rams that is used by the CCMCs to help determine where memory is physically located.
  - The BPRs are the buffers between the CCMC and the rams.
- Figure 6-5 on page 60 is the physical layout of the MTV.
- Figure 6-6 on page 61 is a table showing memory configuration.

Figure 6-5
MTV Block diagram

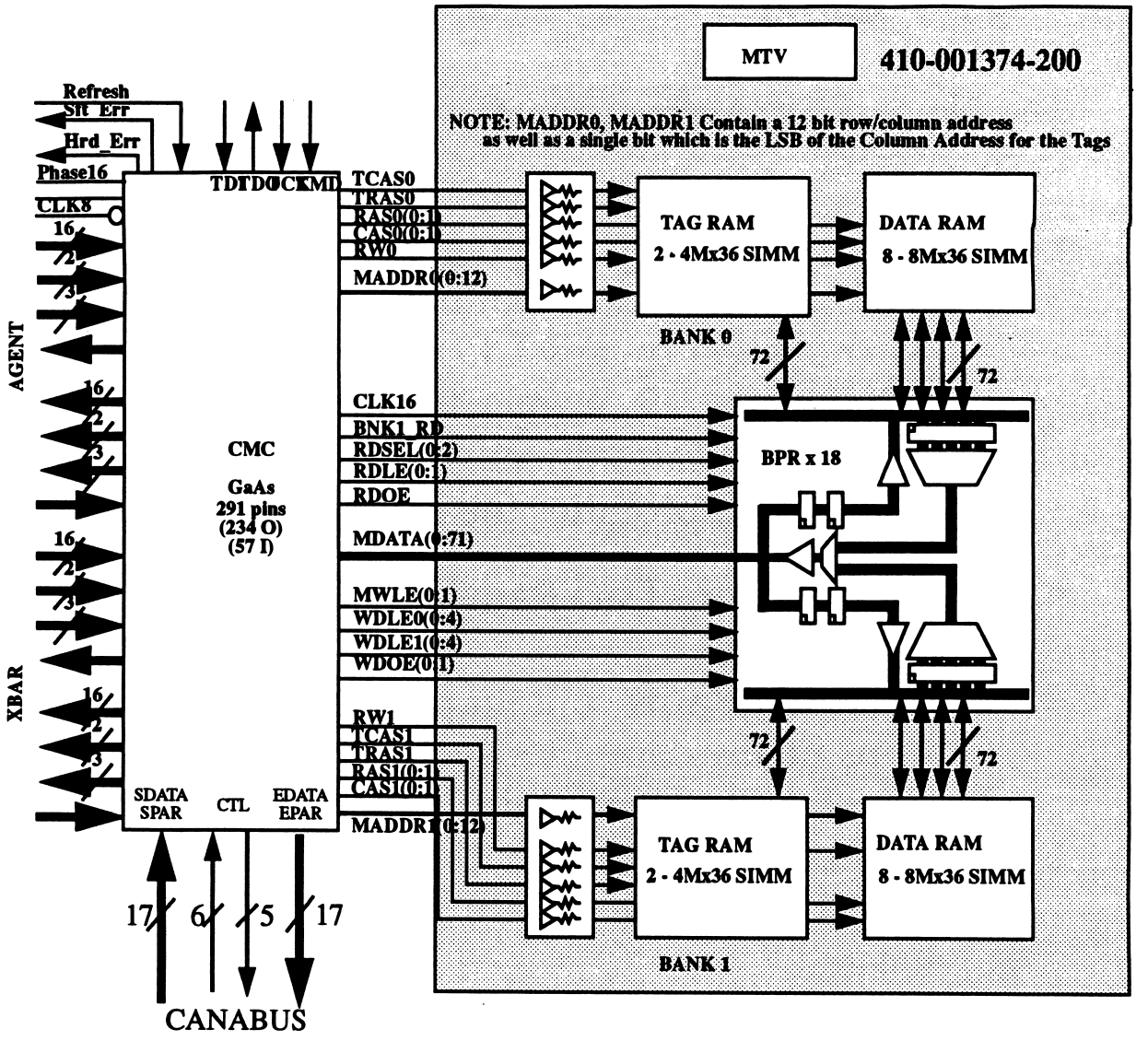
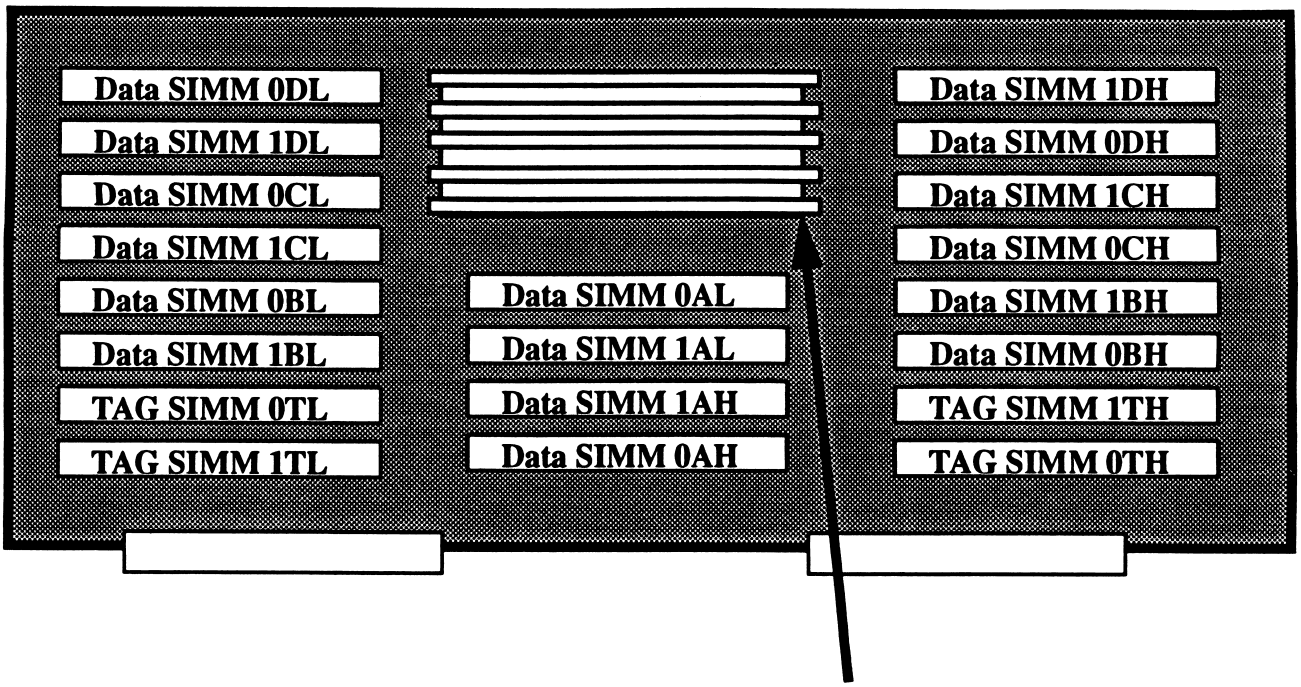


Figure 6-6
MTV Physical Layout

Top View



48VDC to 5VDC Converter Power brick

2 interleaving per board

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	16 DATRAM				512M	550-000060-200
256M		4	16			1024M	550-000061-200
512M			4	16		2048M	550-000062-200
1024M				4	16	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 6-7
Memory configuration table

6.5 Landmarc/ Avalanche

The Avalanche (SIOPII) and Landmarc (SIOP I) are the current I/O boards available for the SPP.

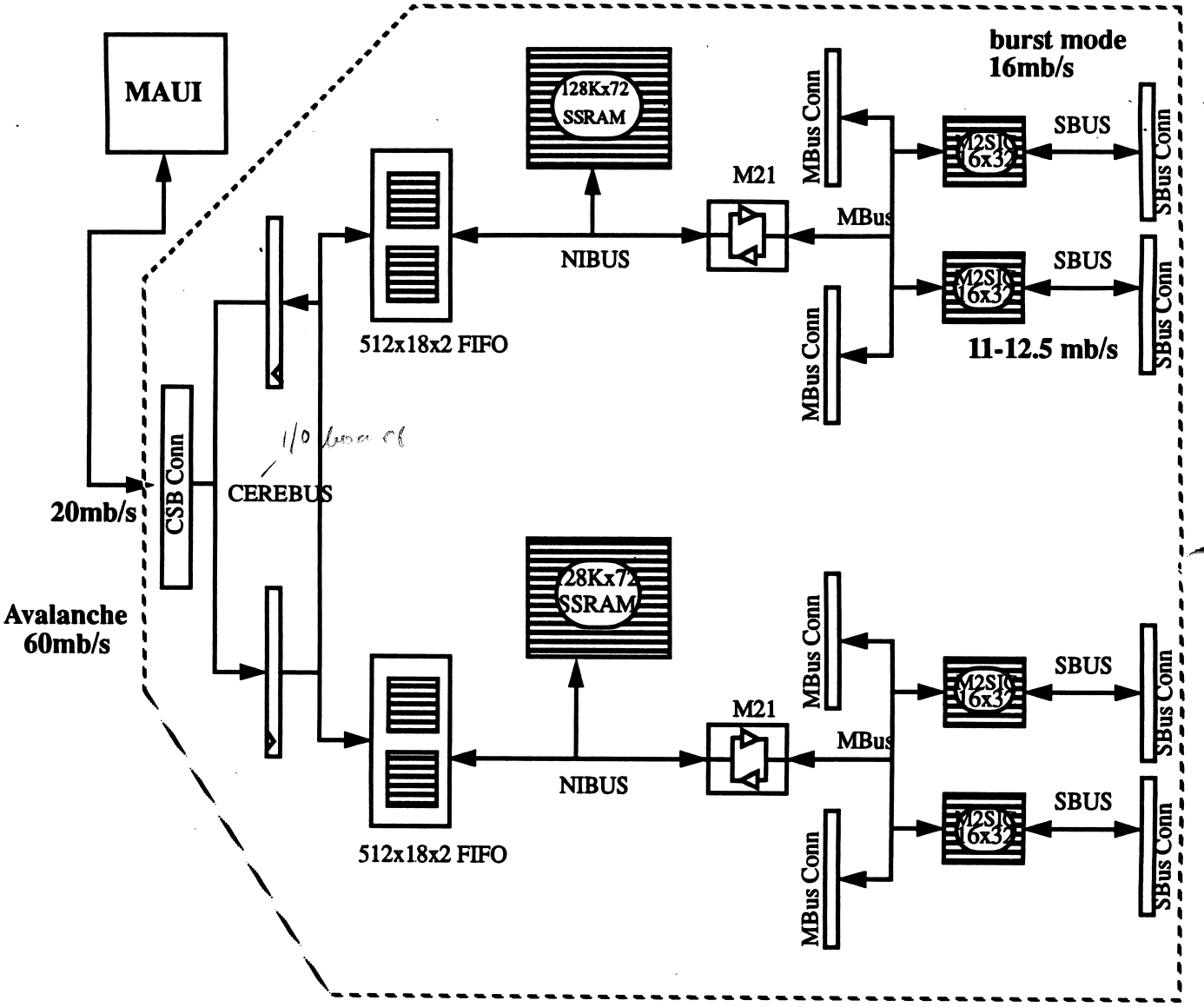
- They communicate to the node via the MAUI.
- The landmarc contains four Sbus ports and four Mbus (NIMbus) ports. Of which only the four sbus ports are usable.
- The avalanche board contains eight ports, of which all are usable.
  - The four Mbus ports are converted to Sbus by the use of **Comsat** boards.
- The Sbus ports use standard Sbus interface controllers.
  - SCSI
  - FDDI
  - and ethernet are currently supported.

Table 3-6 I/O Boards

Board	Part #	Names	# of Ports
SIOP I	2376	Landmarc	4
SIOP I'	4376, 6376	AVIO	4
SIOP II	5376, 7376	Avalanche/ AVIO II	8

Notes: XBAR2s must be present in the XBS position in the spp1000 in order to run the AVIO or AVIO II.

Figure 6-8
Landmarc Block Diagram



The Power System 7

7.1 Introduction

The power system on the SPP-1000 consists of:

- Power brick boards
- Power Bricks
- Power supplies
- Power Chassis
- Power Concentrator Card

7.2 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 do so can damage the SPP.

- Power on sequence:
  - MUon
  - main 48 on

  OR

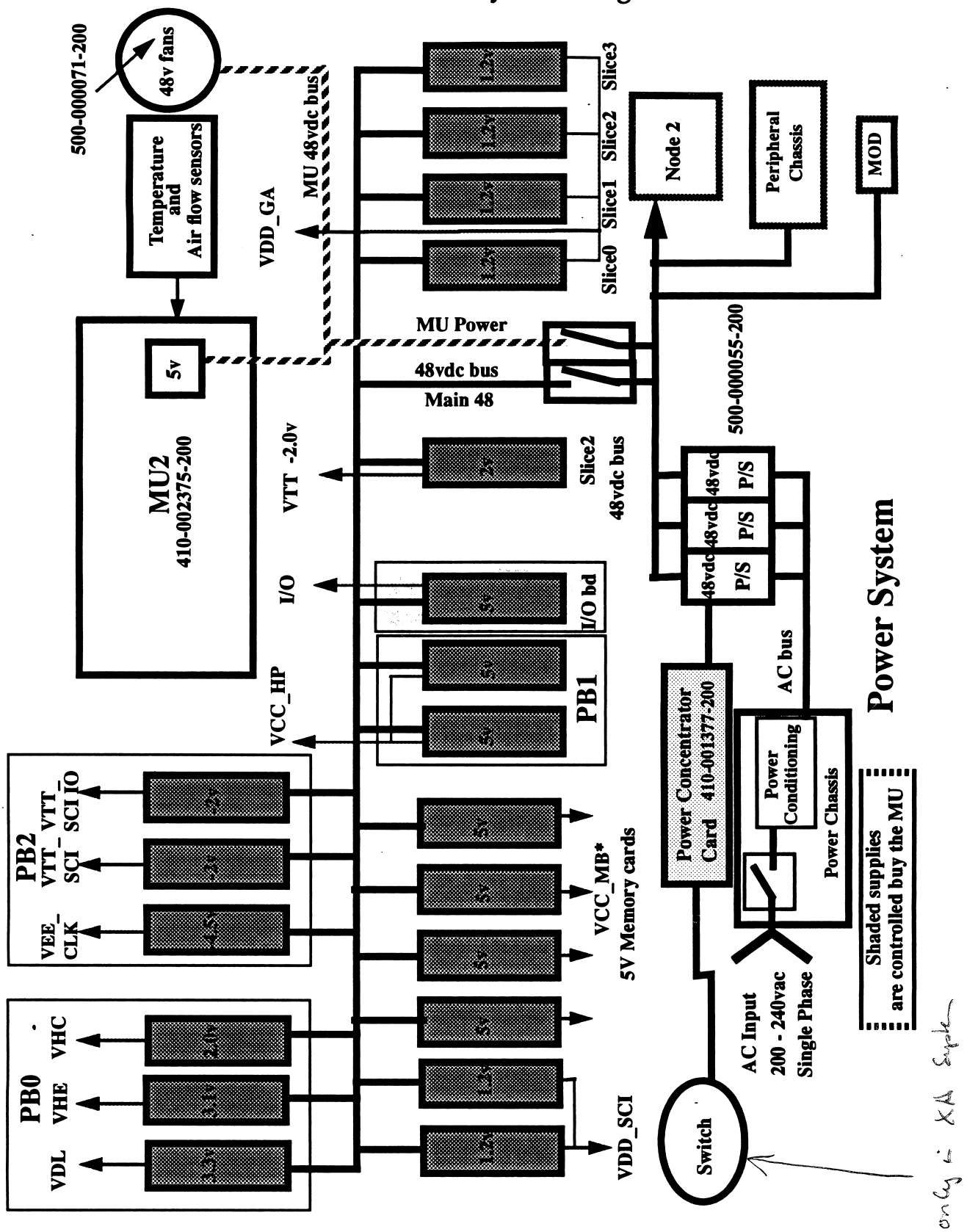
  - keyswitch on (if the SPP was powered down with the keyswitch and/or both MU and Main power switches are already on)

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

  OR

  - pce_util -p off
  - keyswitch off

Figure 7-1
Power Subsystem Diagram



7.3 Power Brick Descriptions

7.3.1 Power Board ~~0~~ 1

- Power Board 1 contains the power brick that supplies **VCC_HP (+5vdc)** to the T-Chip Modules (main power).
- Part Number 410-001378-200

7.3.2 Power Board ~~1~~ 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 gatearrays.
- Part Number 410-001379-200

7.3.3 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

7.3.4 200 Series Power bricks

- 48v to 1.2v 40 amp converter
  - Driver - 200-001049-200.
  - Booster - 200-001049-201.

- 48v to 2v 40 amp converter
  - 200-001047-200

- 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

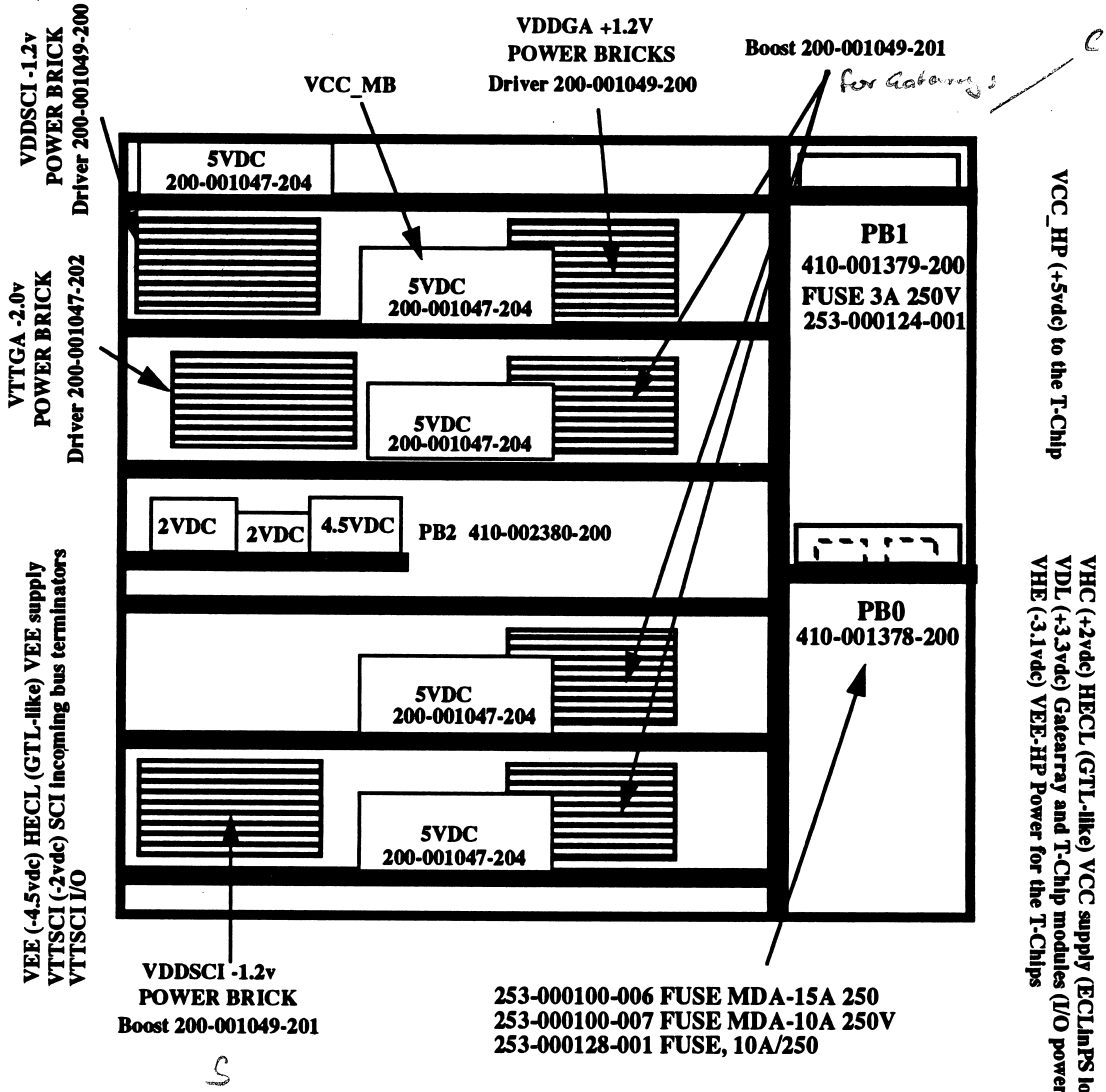
7.3.5 APB Avalon Power Board

- 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.

7.4 Power brick locations

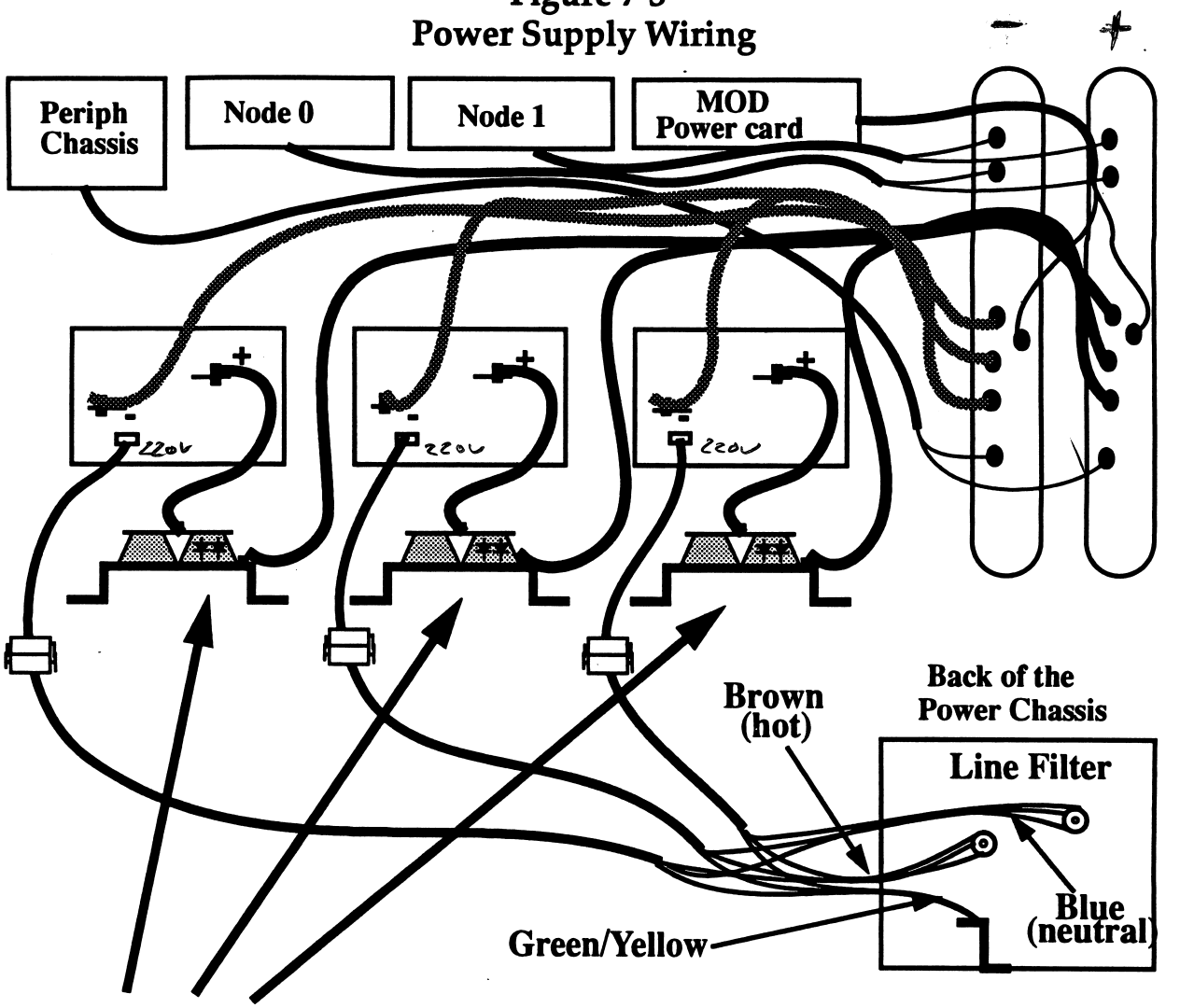
- Brick locations are the same for both the CSB and ASB.
  - Excluding the PB0 and PB1 power brick boards which do not exist on the ASB.

Figure 7-2
Power Brick Locations



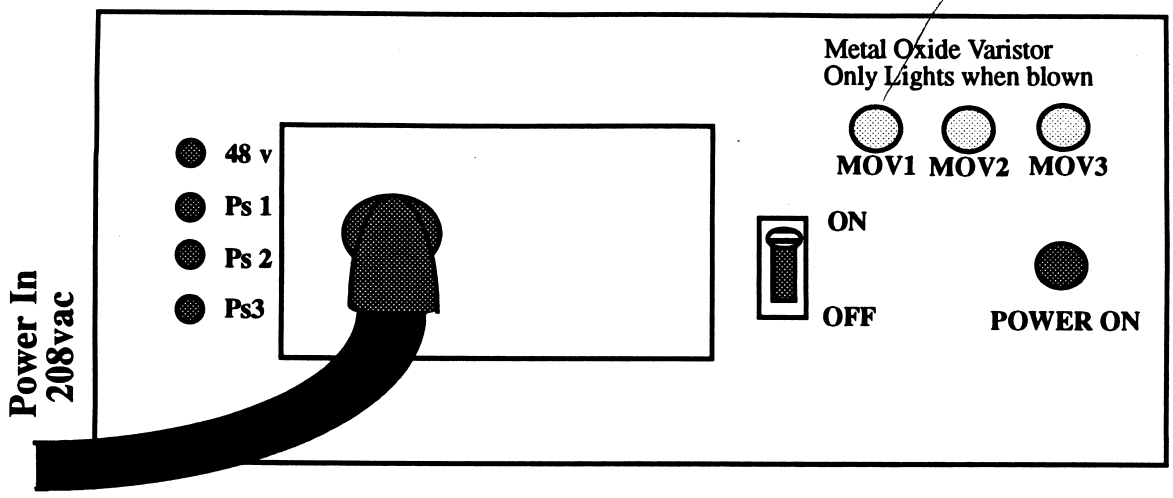
7.5 Power Chassis Wiring

Figure 7-3
Power Supply Wiring



7.6 Power Chassis Front Panel

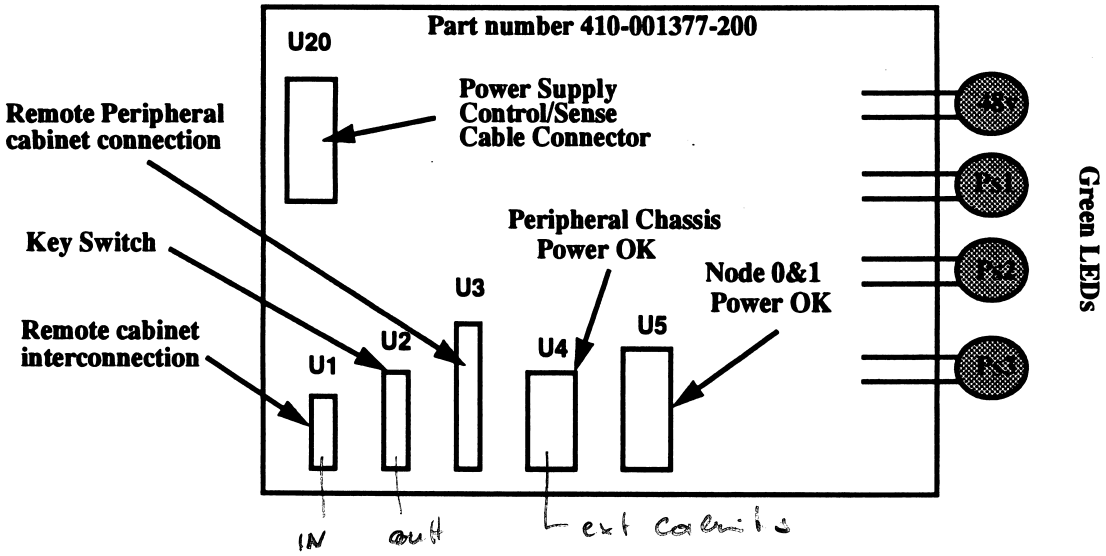
Figure 7-4
Power Chassis Front Panel



(Rear View)

7.7 Power Chassis Concentrator Card

Figure 7-5
Power Concentrator Card



- **Remote peripheral cabinet connection:** Connect to external peripheral cabinets for remote power up.
- **Key switch:** Cables toggle switch on front of SPP cabinet (located in same area as the LCDs) to control 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 the nodes.

MEMORY 8

8.1 Physical memory divisions

The SPP-1000 divides it's 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 7-6
Node Physical Address Space Partitioning

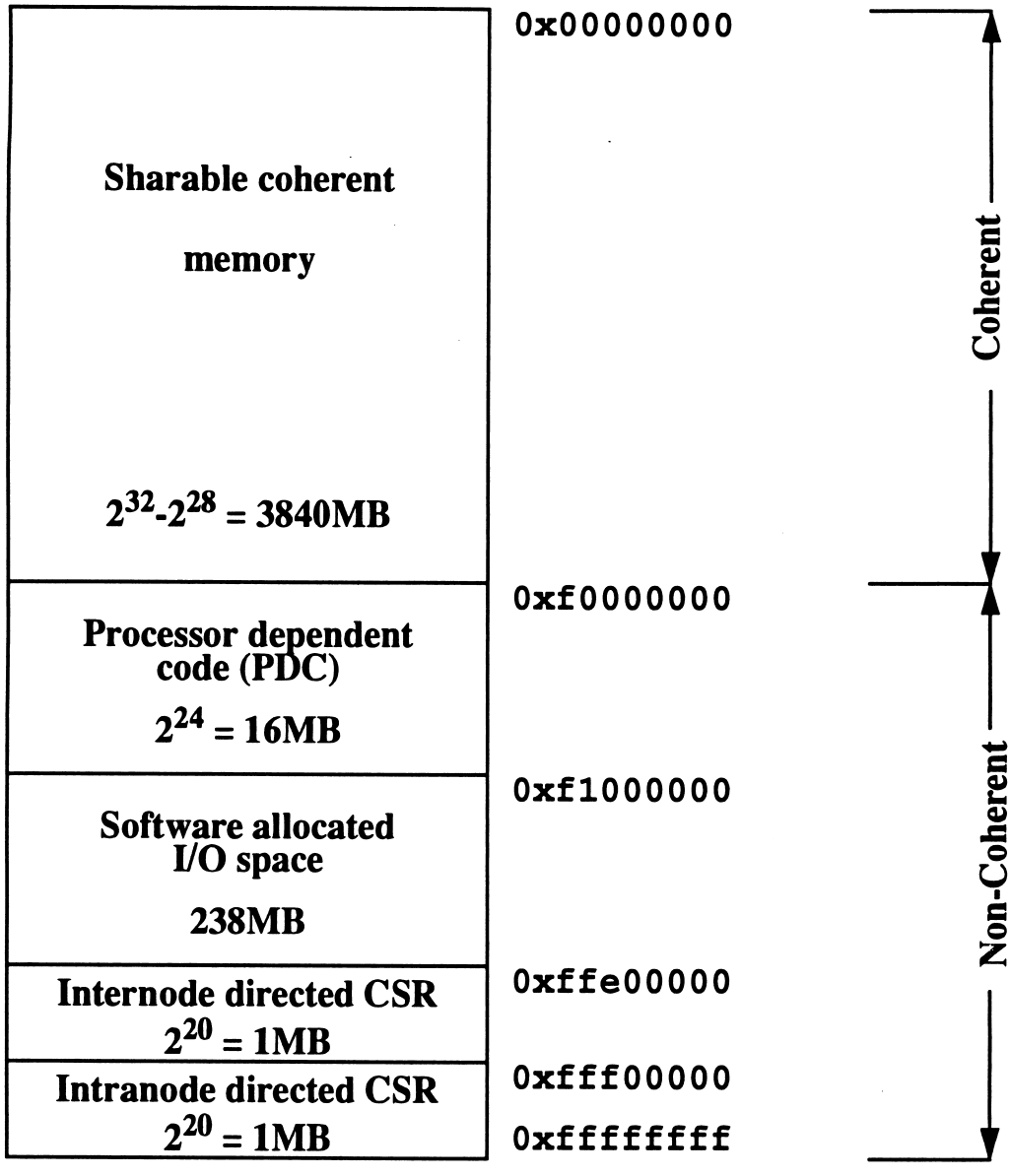
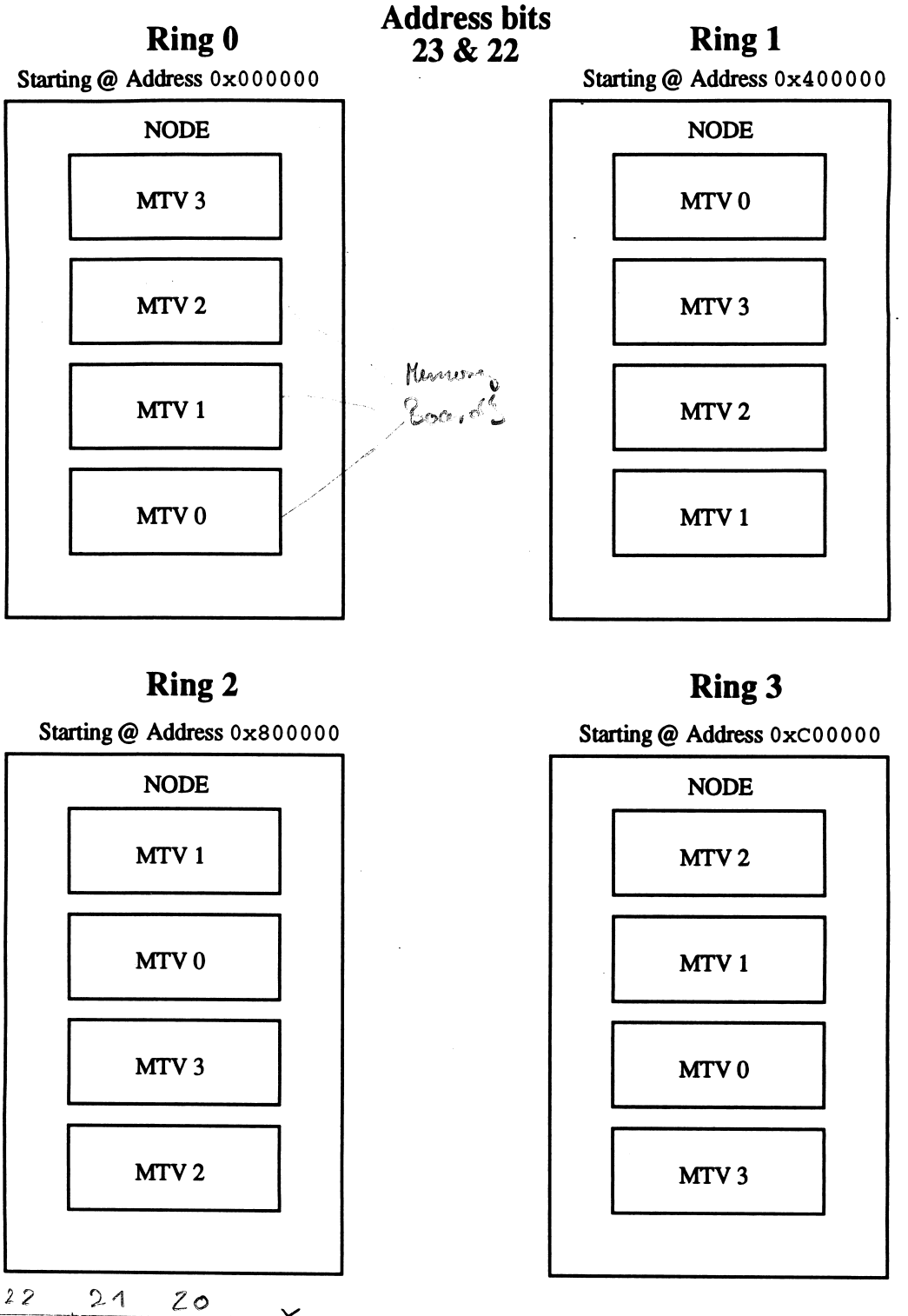


Figure 7-7
Physical Memory Addressing with Ring Bits



Internal Communications 9

The communication between functional blocks in the SPP-1000 is done by a complex system of transactions. These transactions are accomplished with a **packet system**. In this section, packet descriptions and formates will be detailed.

9.1 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.

9.2 Packet Types

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

9.2.1 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)

9.2.2 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

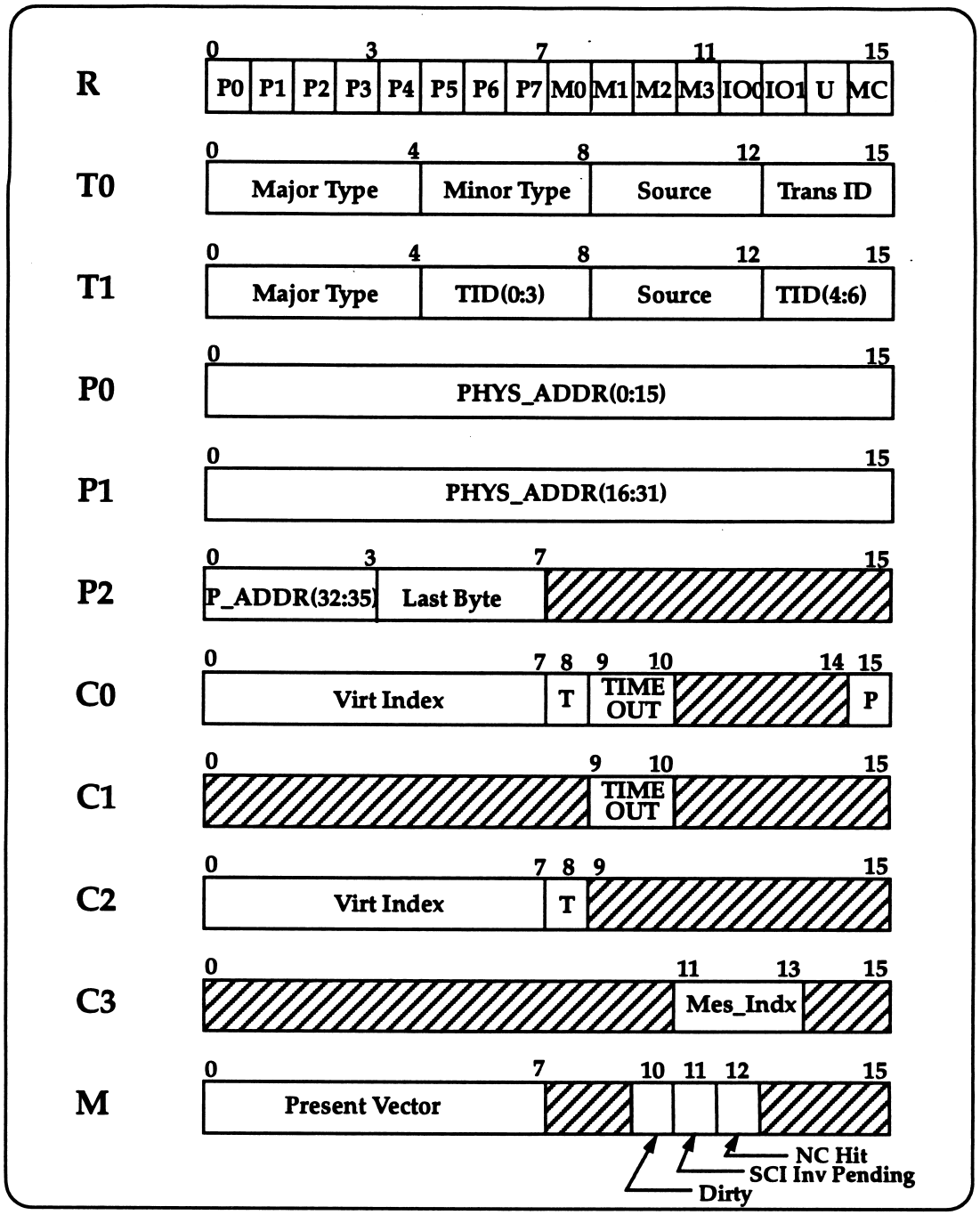
9.3 Packet Structures

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 8-8
Packet Word Format



Packet word formats (Continued)

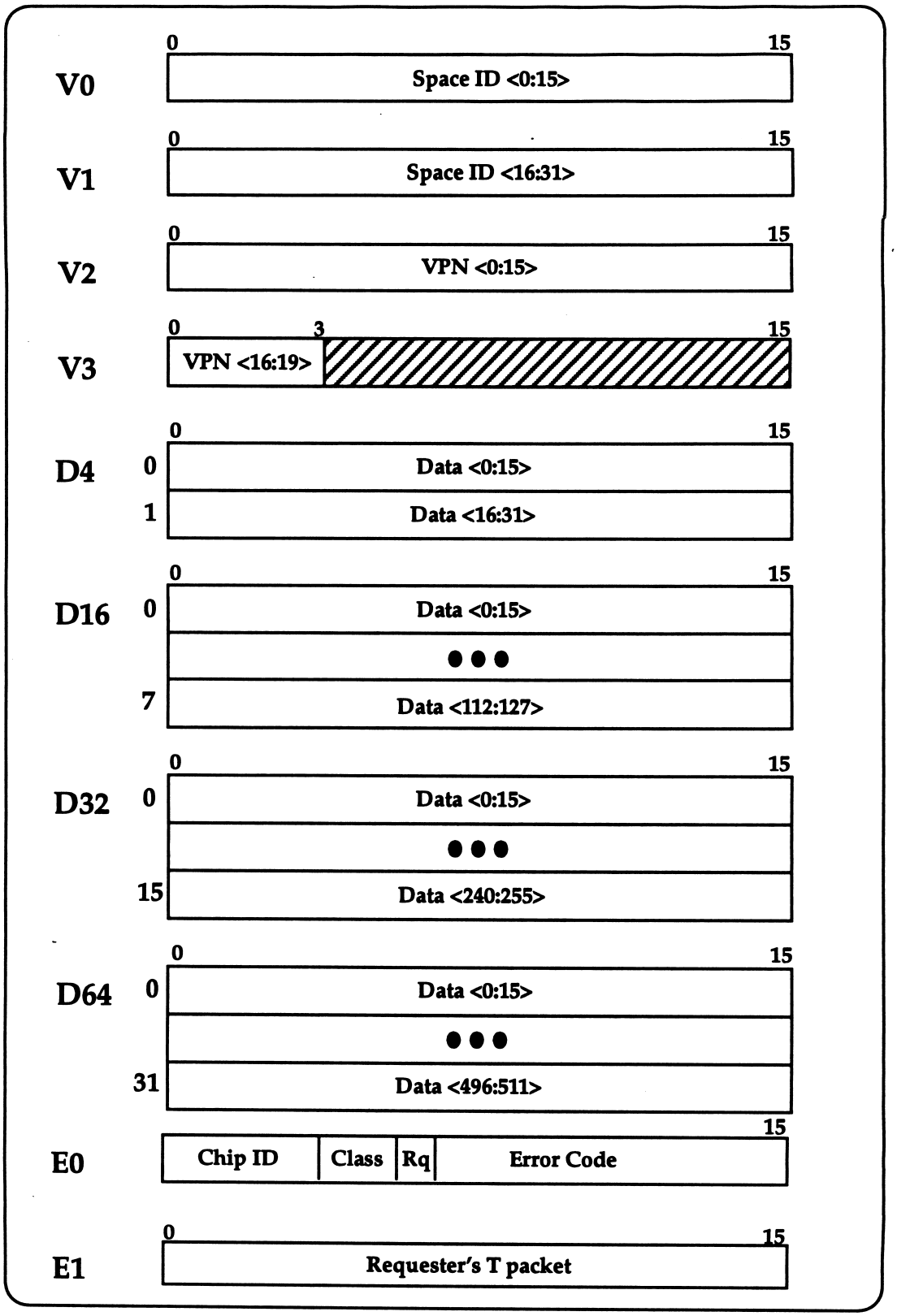


Table 9-1 Typical packet formats (request packets)

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 9-1 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

Meaningful Optical Display 10

10.1 Introduction

The **MOD** lights (**M**eaningful **O**ptical **D**isplay) are located on the front, top and back of the XA system chassis.

- They are programmable by the customer.
  - A set of C libraries are provided for the customer's convenience.
  - 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.

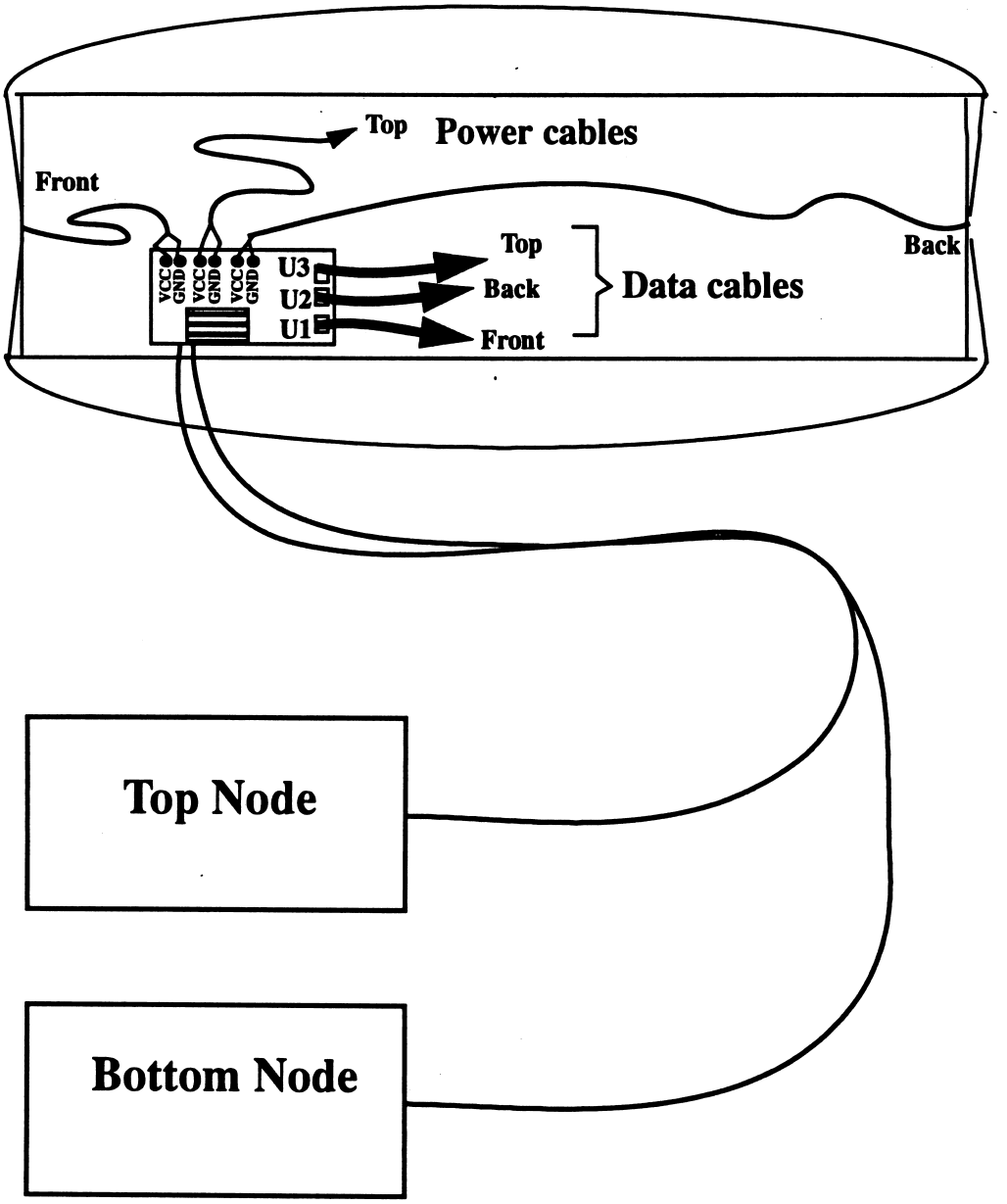
10.2 The MOD light controller

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.

10.3 MOD light cable routing

Figure 9-1
Mod light power and data cable routing
Top View of the cabinet

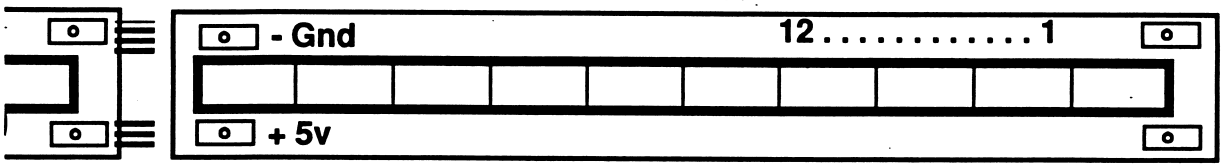


10.4 MOD Board Strapping

Figure 9-2
MOD Board Strapping Diagram

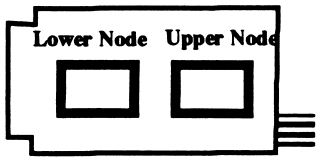
Total of 4 boards for each side
Plus Status board on front

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



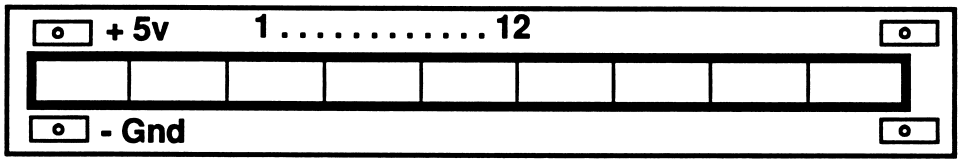
	Jumper Settings			
Upper (Board 3)		8-7		
Upper Middle (Board 2)		8-7		2-1
Lower Middle (Board 1)		8-7	5-4	
Lower (Board 0)	12-11	8-7	5-4	2-1

Node Status board



410-001434-200

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

10.5 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

Gate Array Removal and Replacement 11

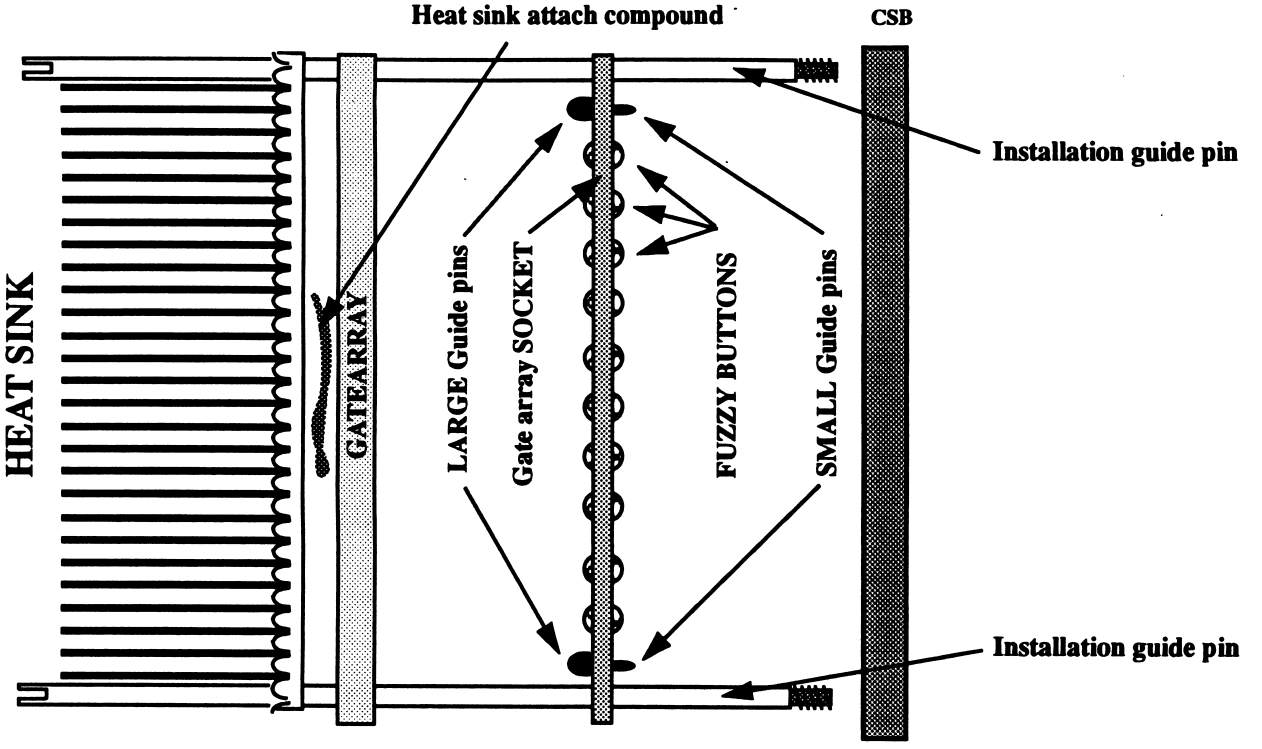
11.1 Field Gate Array Replacement

- Used for:
  - Part upgrades
  - Trouble shooting
- See the figure on the next page for more details

11.2 Removal and Replacement Parts

- Things you will need to correctly replace a down-reved gate array in the field are:
  - #1 Phillips head long shank screw driver
    - magnetic tip it *VERY* helpful.
  - The new revision gate array (with it's heat sink attached).
  - Installation guide pins (Part number 315-000205-500).
  - Gate array socket.

Figure 10-1
Gate Array Removal and Replacement



11.3 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 to stiff but firm enough to remove contamination).

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

11.4 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.

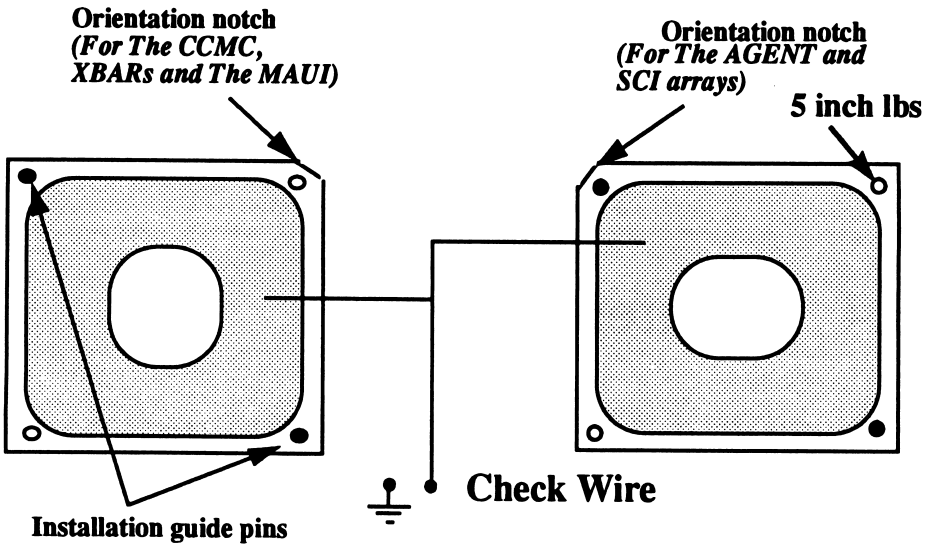
- Torque should be checked and reset to 7 inch lbs at installation and every 90 days.
  - This problem is being reviewed, future torque specs may increase in order to resolve this problem.

11.5 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.

Figure 10-2
Gatearray and Socket orientation



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.

I/O 12

fW = 15 devices
Convex only: 10 devices

12.1 Boards

- The boards in the peripheral chassis are:
  - **CIOBP** - Convex IO backplane (410-001438-200)
  - **CIOIR** - Convex IO Right address card (410-001440-200)
  - **CIOIL** - Convex IO Left address card (410-001441-200)
  - **CIOPC** - Convex IO Power Controller (410-001442-200)
  - **Ancot** - SCSI Differential to Single ended converter (550-000423-200)
  - **CIODA** - Interface card for Disk Drive to XA IO chassis 410-001439-200

12.2 Peripheral Configurations

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

12.3 XA

12.3.1 High Performance

Figure 11-1
The High Performance Peripheral chassis

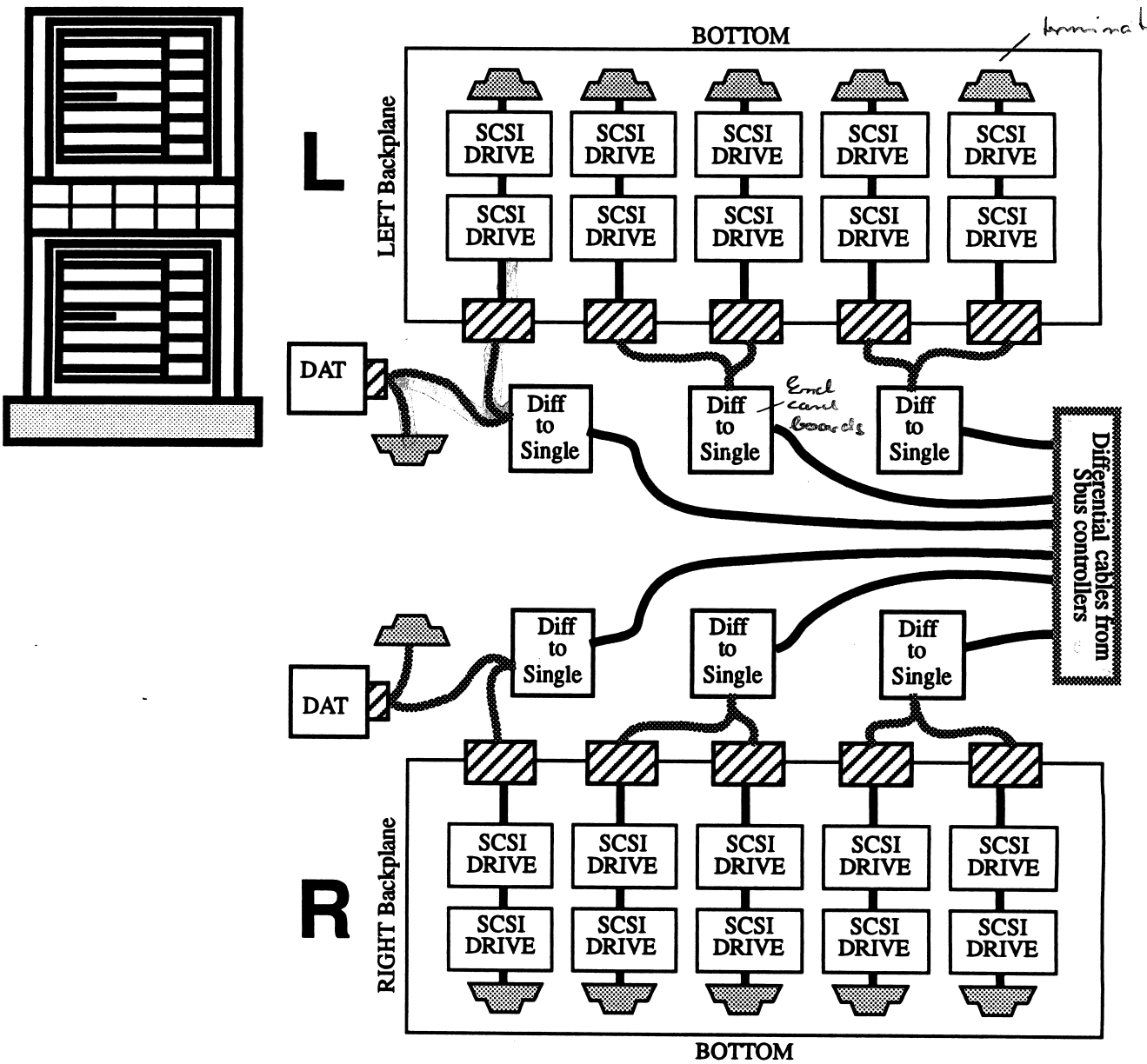
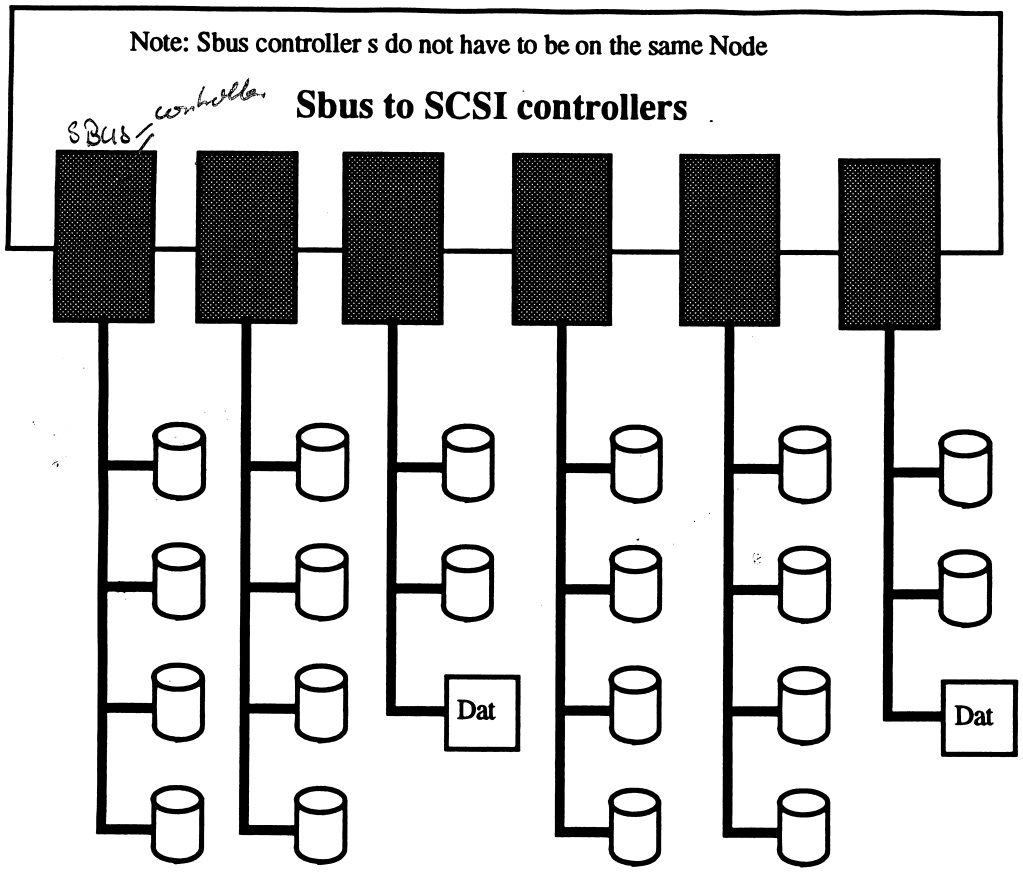


Figure 11-2
The High Performance Block Diagram



12.3.2 High Capacity

Figure 11-3
The High Capacity Configuration

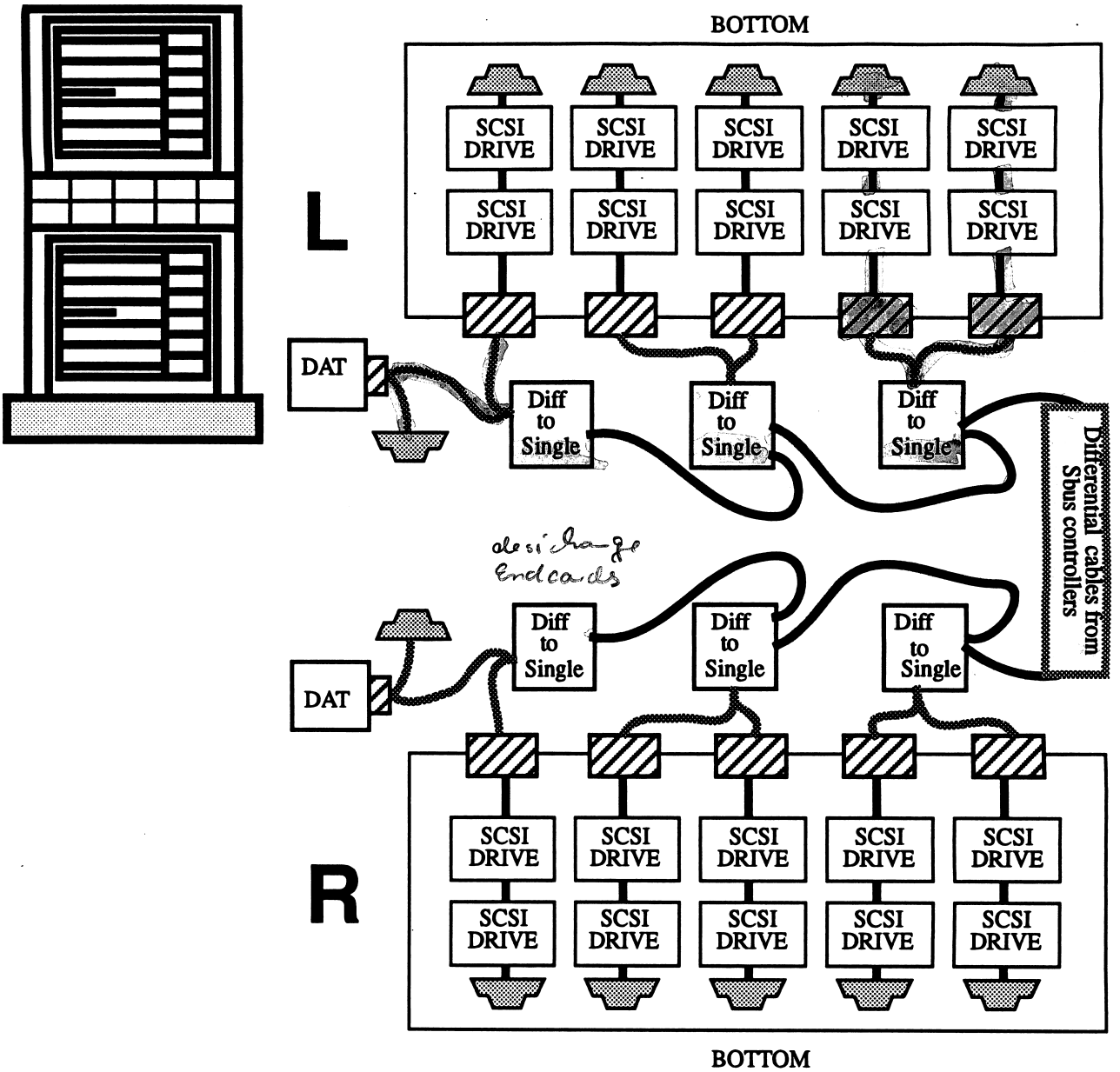
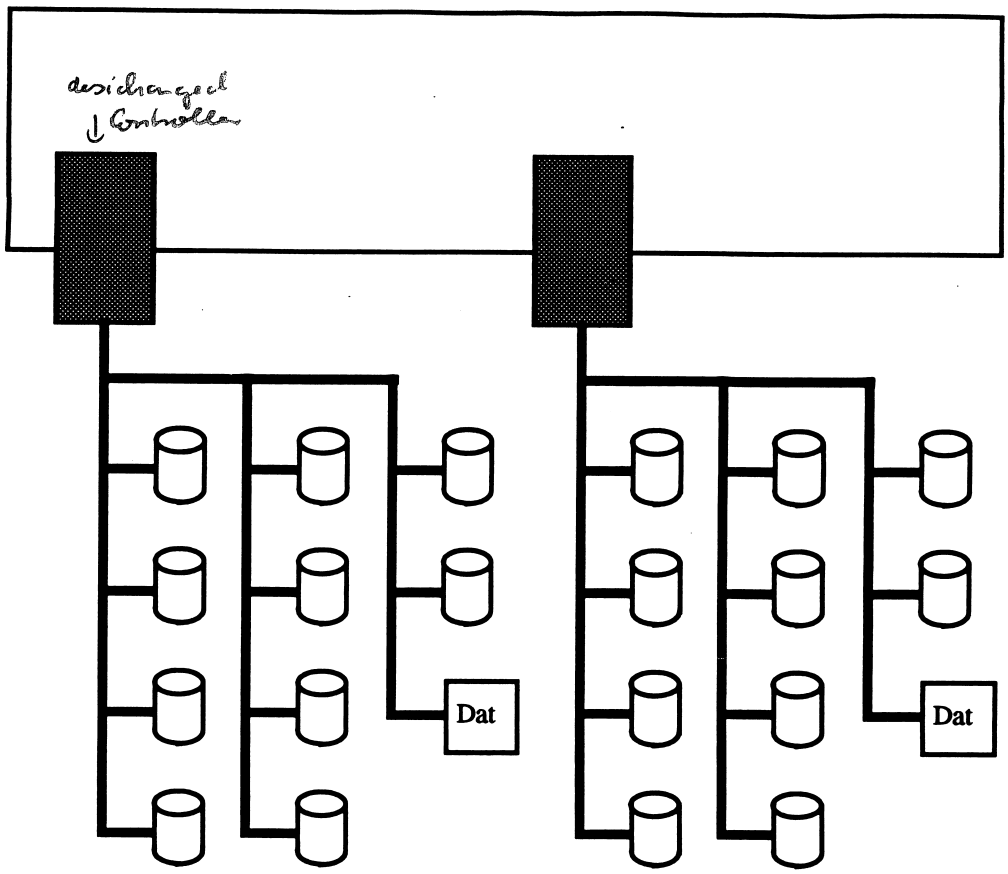
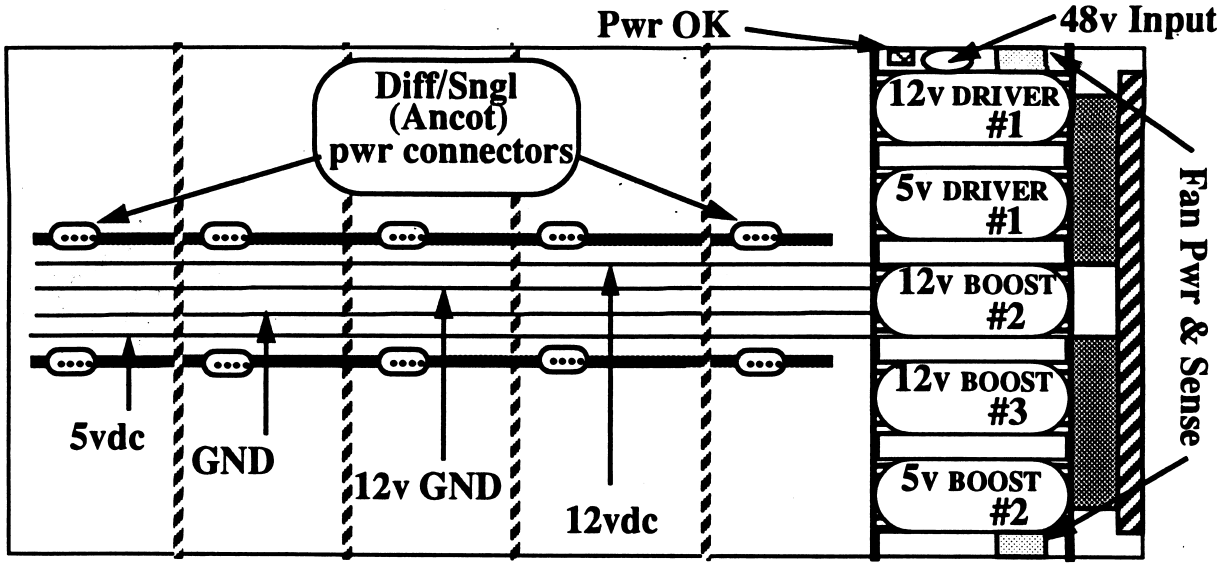


Figure 11-4
The High Capacity Block diagram



12.3.3 Power system

Figure 11-5
Peripheral cabinet power subsystem



CIOPC 410-001442-200 IO Chassis Power Controller

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

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

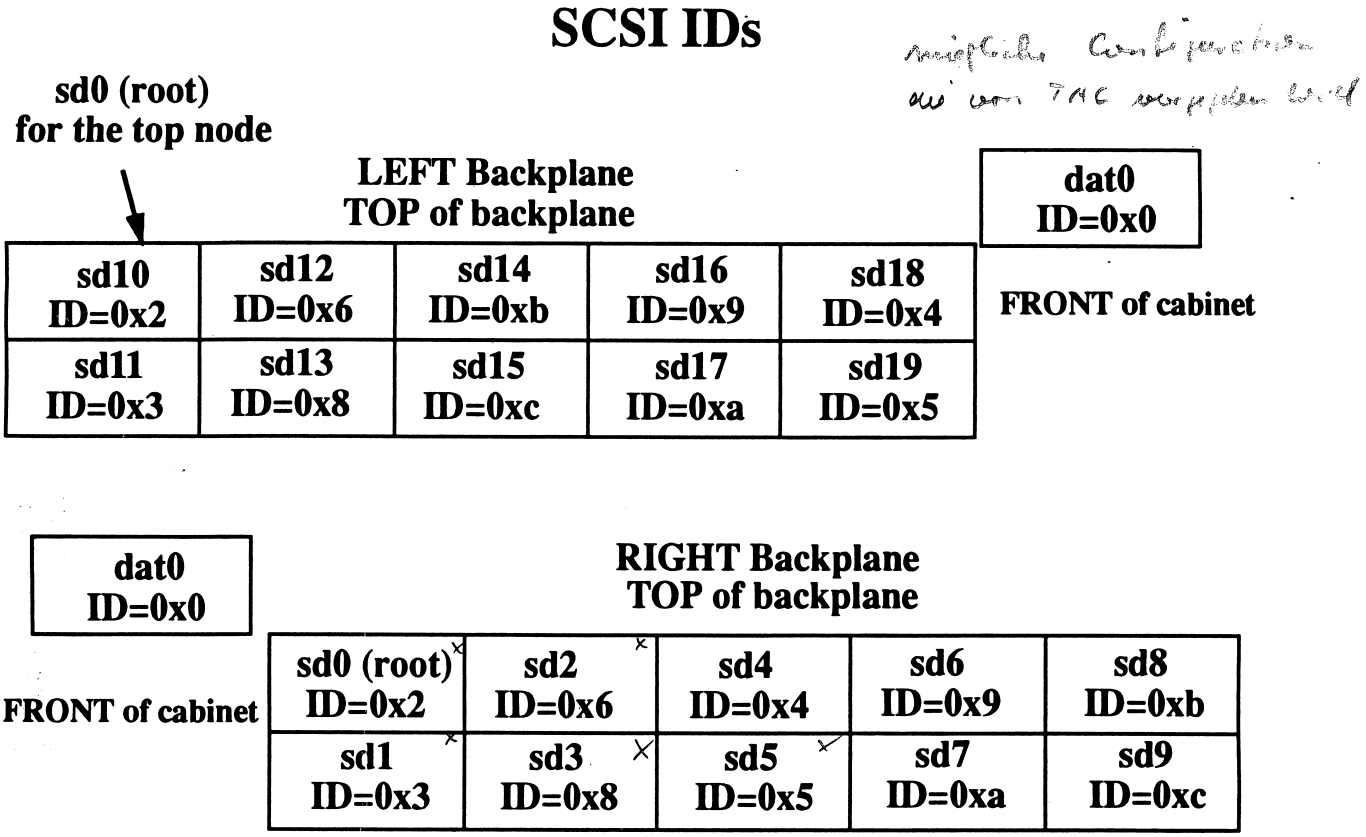
410-003443-200 BD ASSY, CIOPB, 5V, BOOSTER

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

Di Power supply con I/O Chassis point con Power panel ri L

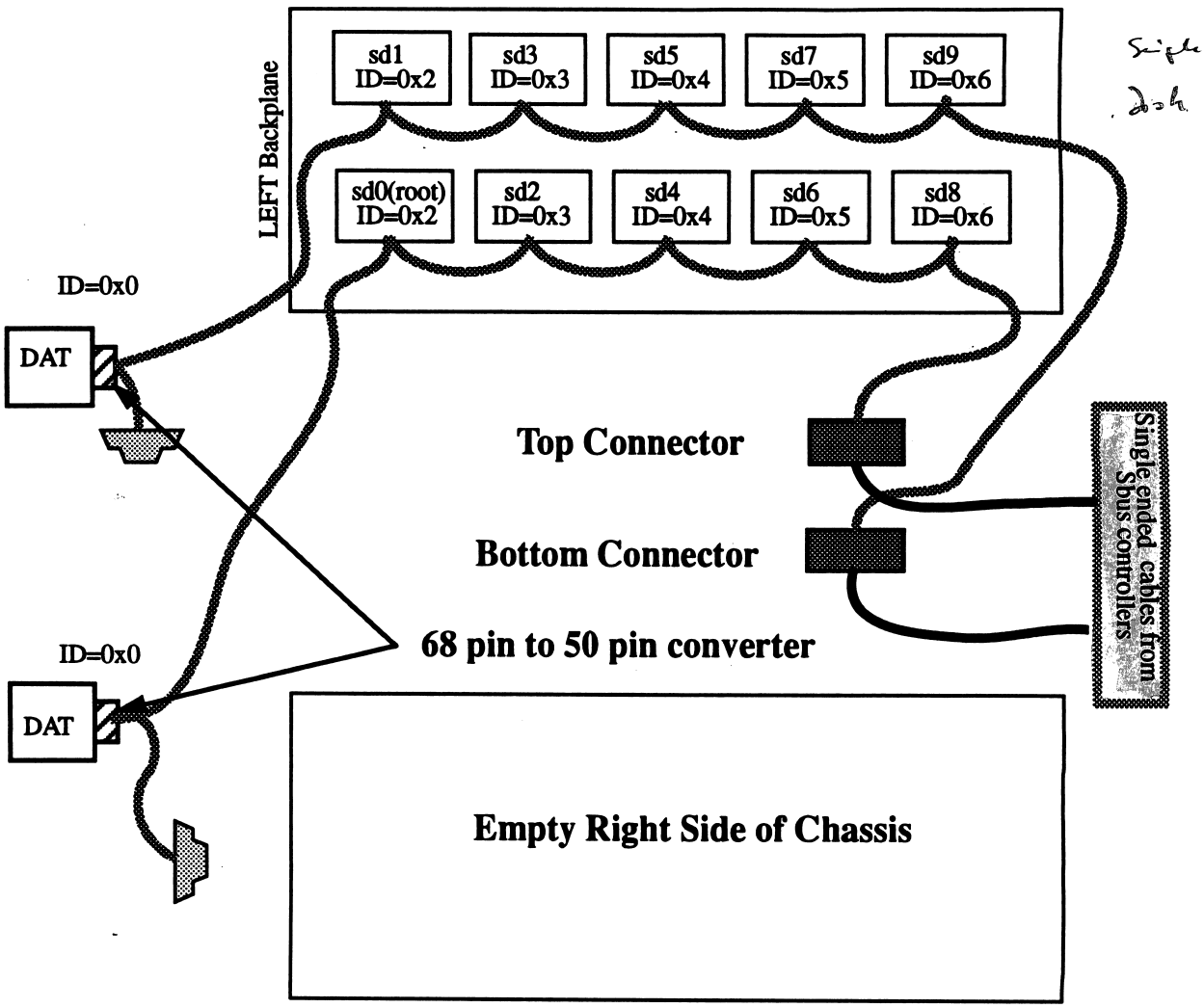
12.3.4 Slot Addresses

Figure 11-6
High Density SCSI ID's



12.4 CD Configuration

Figure 11-7
CD cable diagram

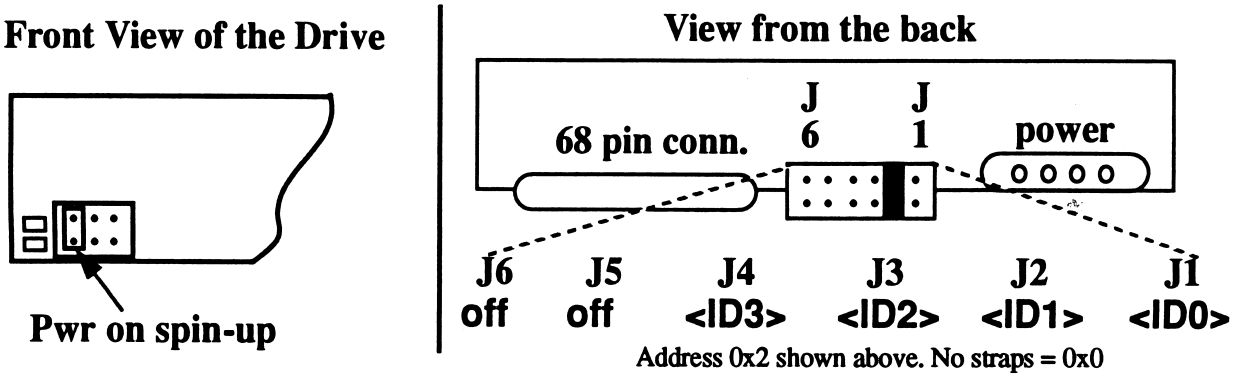


12.5 Disk Drives - Strapping

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

12.5.1 DEC (2 GIG)

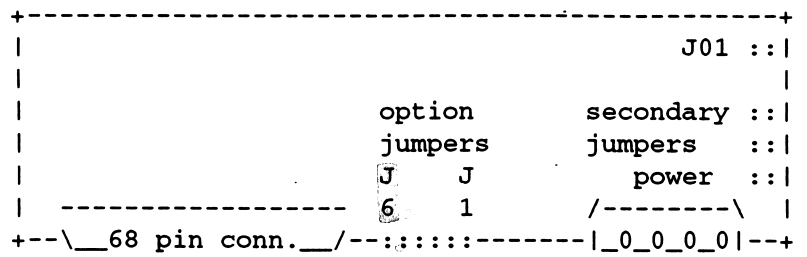
Figure 11-8
Disk Drive Strapping (DEC DSP3210W 2)



Address 0x2 shown above. No straps = 0x0

12.5.2 Segate (4 GIG)

- Option connector:



Jumper Settings on Option Connector

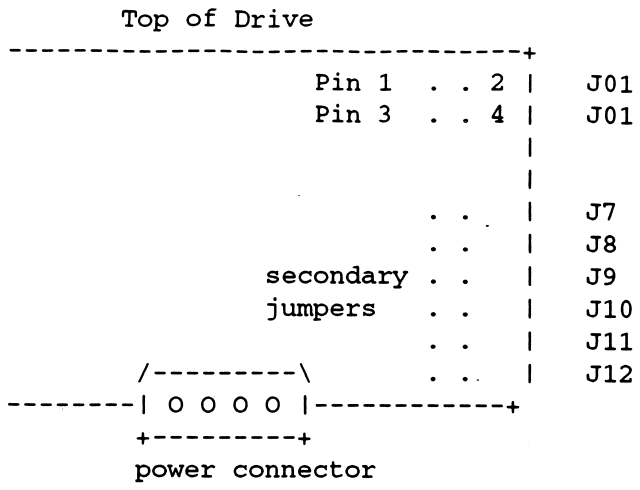
J6	J5	J4	J3	J2	J1
Off	Off	<ID3>	<ID2>	<ID1>	<ID0>

  - 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 staring 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.

12.5.3 Dat Drive

- 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

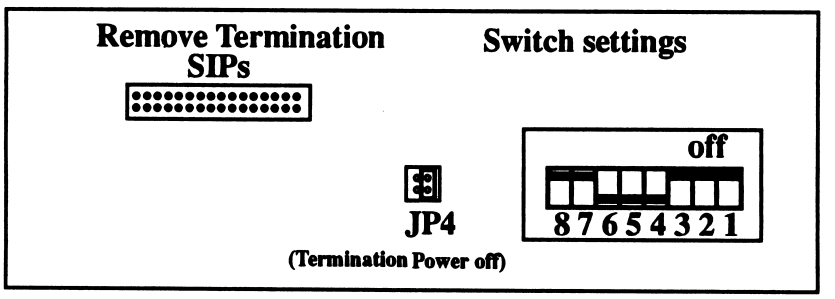
Switches S1, S2, and S3 control the SCSI device address.

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:

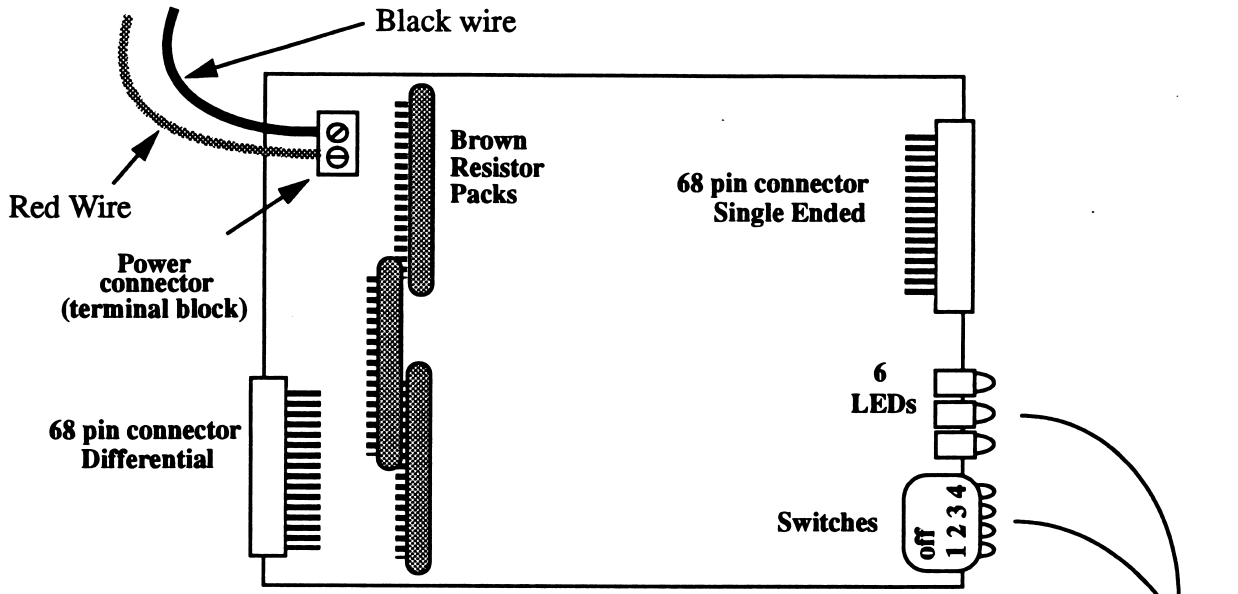
Switch	S4	S5	S6	S7	S8
Setting	ON	ON	OFF	OFF	OFF
Meaning	MRS	Parity	Compression	reserve	Self-test
	On	On	Enabled		

Figure 11-9
DAT Drive strapping



12.5.4 Ancot Card

Figure 11-10
Differential to Single Ended Converter Card



LED Meanings

SE Termpower (green)	Differential Termpower (green)	Power OK (green)
SE Active Terminator (orange)	Differential Terminator (orange)	Busy/Active (orange)

	Switch settings	Switch settings for a board in a chain	Switch settings the LAST board
4	Terminator for Diff Side	up	down
3	Terminator for SE Side	up	up
2	Termpower for Diff Side	up	up
1	Termpower for SE Side	down	down

12.6 IO Address Reference Sheet (nvalias format)

Addresses as seen in the OBP window:

- *Landmarc in node 0*
  /landmarc@0
- *Landmarc in node 1*
  /landmarc@10000
- *Landmarc in node 15*
  /landmarc@f0000
- *Landmarc node 1unit 0*
  /landmarc@10000,ffec0000
- *Landmarc node 1unit 1*
  /landmarc@10000,ffed0000
- *Landmarc node 0 unit 0 sbus slot 0*
  /landmarc@0,ffec0000/sbus@f,fcffff00
- *Landmarc node 0 unit 0 sbus slot 1*
  /landmarc@0,ffec0000/sbus@f,fdffff00
- *Landmarc node 0 unit 1 sbus slot 0*
  /landmarc@0,ffed0000/sbus@f,fcffff00 (slot 2 - feffff00
- *Landmarc node 0 unit 1 sbus slot 1*
  /landmarc@0,ffed0000/sbus@f,fdffff00 (slot 3 - ffffff00)
- *Landmarc node 0 unit 0 sbus slot 0 Fast/Wide SCSI*
  /landmarc@0,ffec0000/sbus@f,fcffff00/Convex,afws@1,10000
- *Landmarc node 0 unit 1 sbus slot 0 FDDI CRESENDO*
  /landmarc@0,ffed0000/sbus@f,fcffff00/CRES,cddi@1,400000
- *Landmarc node 0 unit 0 sbus slot 0 Fast/Wide SCSI SCSI ID 0x2 partition a*
  /landmarc@0,ffec0000/sbus@f,fcffff00/Convex,afws@1,10000:narrow/sd@2,0:a

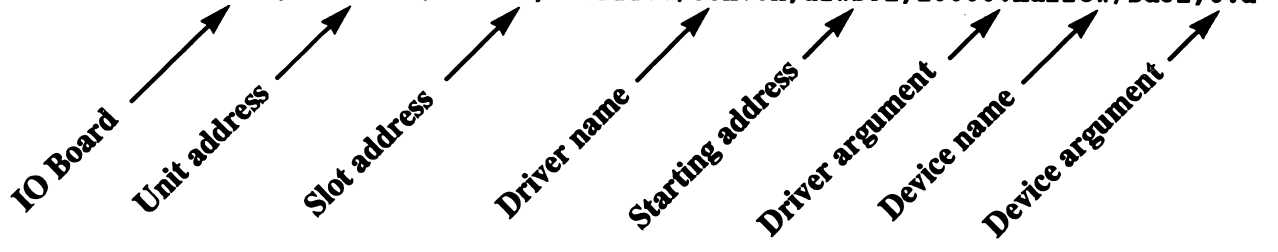
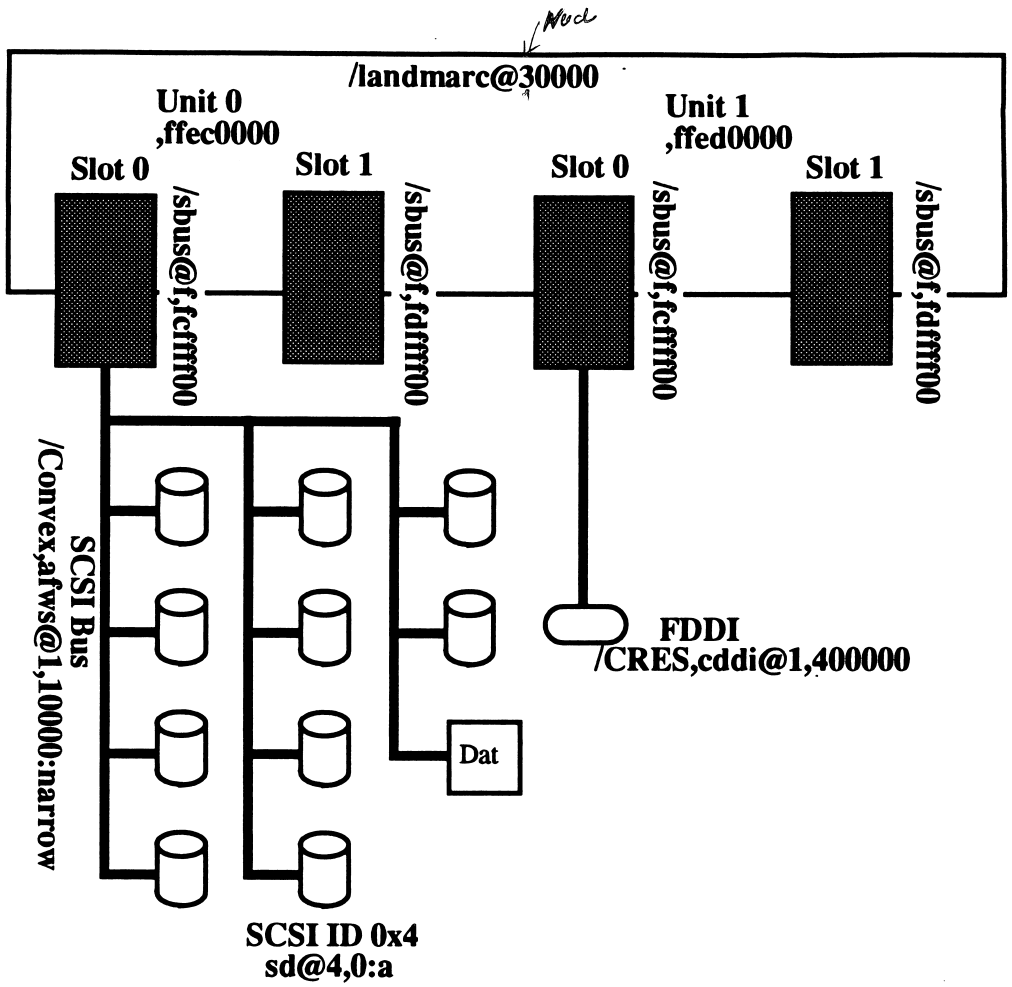


Figure 11-11
IO Address Figure



12.7 OBP Setup Parms for IO Devices (mkmap)

12.7.1 Dat Drive Examples

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

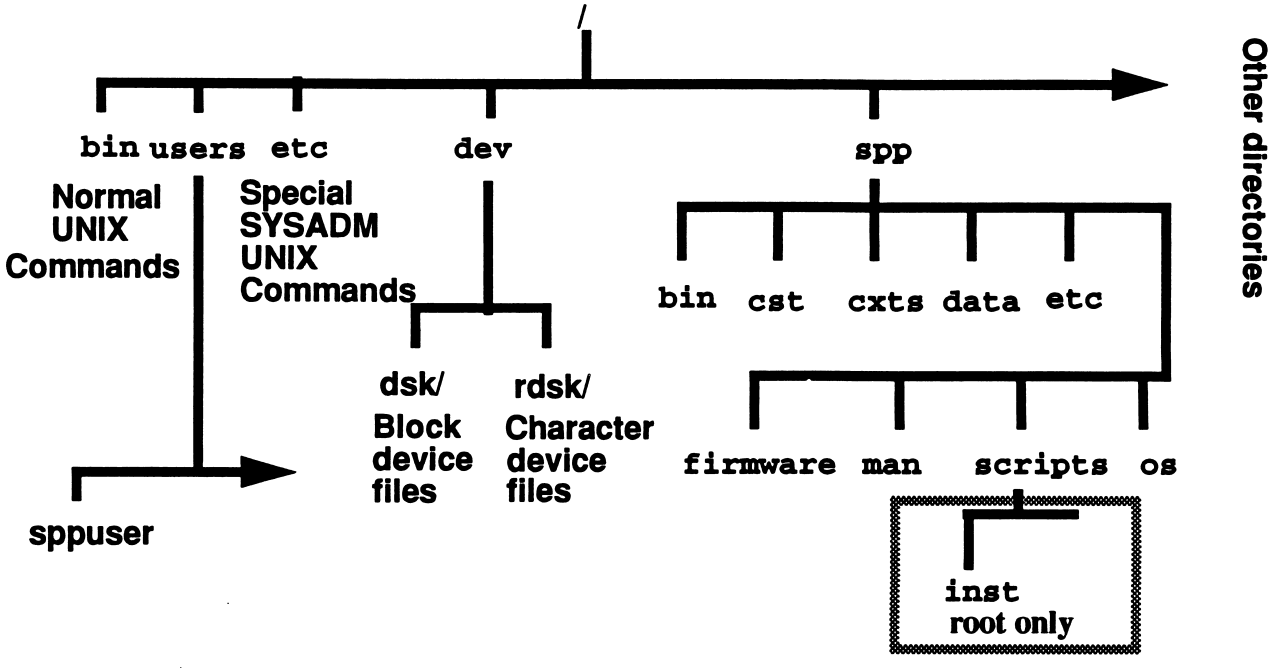
- 1 Dat on node 0
  - **0 node**
  - **mkmap 0 /landmarc@0,ffec0000/sbus@f,fcffff00/Convex,afws@1,10000/st@0,0**
- 1 Dat on node 1
  - **1 node**
  - **mkmap 1 /landmarc@10000,ffed0000/sbus@f,fdffff00/Convex,afws@1,10000/st@0,0**
- 2 Dats on node 2
  - **2 node**
  - **mkmap 2 /landmarc@20000,ffec0000/sbus@f,fcffff00/Convex,afws@1,10000/st@0,0**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

12.7.2 FDDI Example

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

Test Station 13

13.1 The file System Tree



13.2 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

13.3 Useful Test Station Commands

13.3.1 Backup - In SPP-UX

- Backing up the complex's file systems from the test station:
  - Perform the backup as root.
  - 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 & ← fix ab 3.2.

- Use fbackup as described in the SPP-UX System Administration Guide.

13.3.2 Backup - HP UX (test station files)

- Same as in SPP-UX, **fbackup**.
  - EX: /etc/fbackup -f /dev/rmt/3m -0 -i /
  - The "-e" option can be used exclude directories.

13.3.3 Restore

- Perform as root
  - Install backup tape

/etc/frecover -f /dev/rmt/3m -i /

13.3.4 Creating a Bootable Recovery Tape - HP-UX

To make a tape the test station can boot off of:

- Performed as root.
  - Install a blank tape in test station dat drive.

/etc/mkrs -q -v -f /dev/rmt/3m -r /dev/rdsk/c201d6s0 -s -m 700

13.3.5 Software Install

- Perform as root.
- Install the software tape, type:
  - **/etc/installsw -n**

13.3.6 Test Station Configuration

- /usr/bin/sam -display [test station name]

13.3.7 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**

13.3.8 Boot SPP-UX From the Dat Drive

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

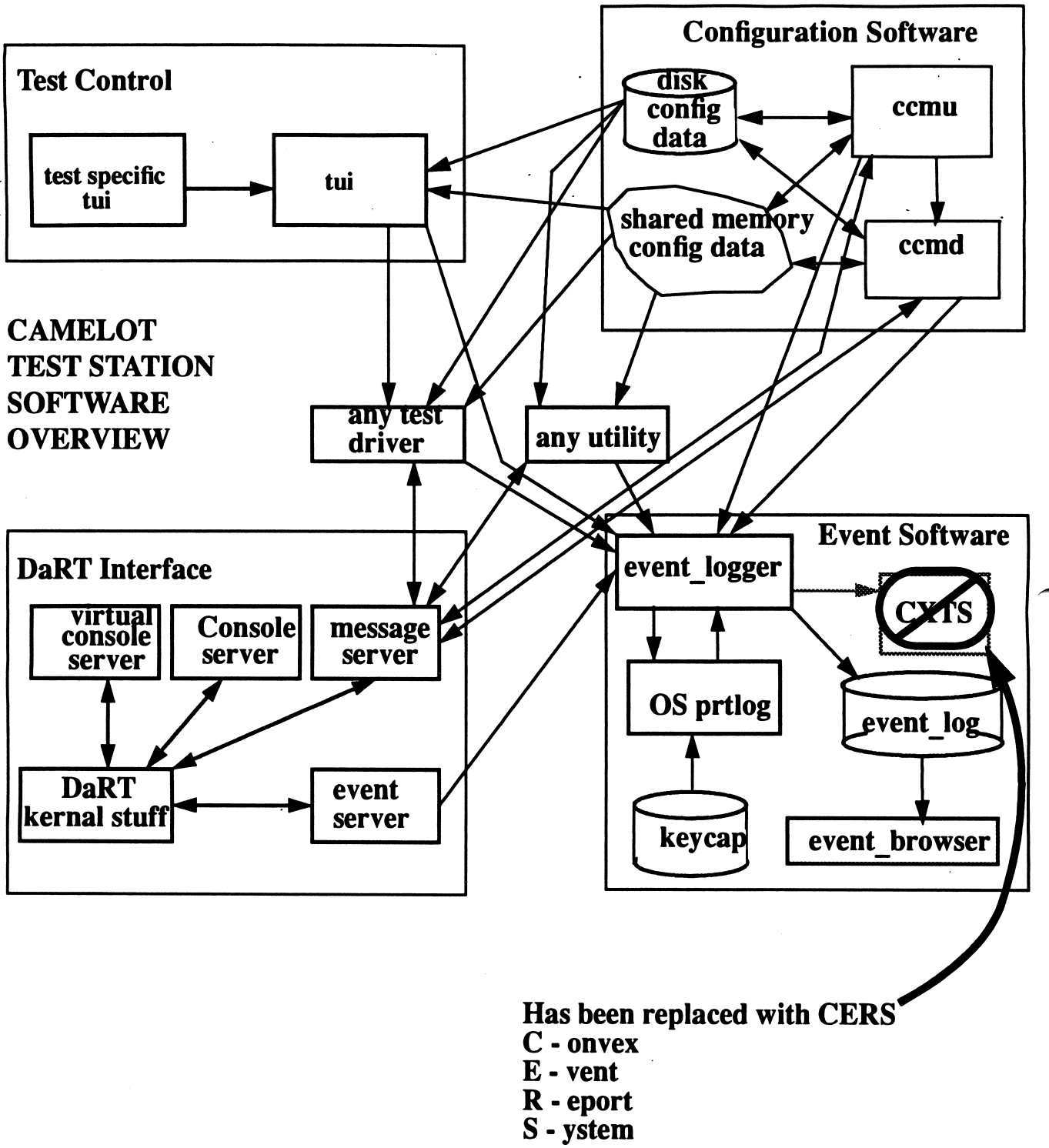
Diagnostics & Utilities 14

14.1 Test Station SW Introduction

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 14-1
Test Station Software Overview



14.2 Software/ Firmware Versions.

14.2.1 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 67fd 2b05
0100000
```

14.2.2 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    "CONVEX Version Info:    2.1"
```

(revision 3.X will also give the revision of the primary loader)

- 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**

14.2.3 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/sppring**

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.

14.2.4 SPP-UX Versions.

- At the spp-ux OS prompt use the following command:
  - **vers /os/mach**
  - **vers /os/server**

14.3 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 deamon.
- 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
  - **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
- **memsize** 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

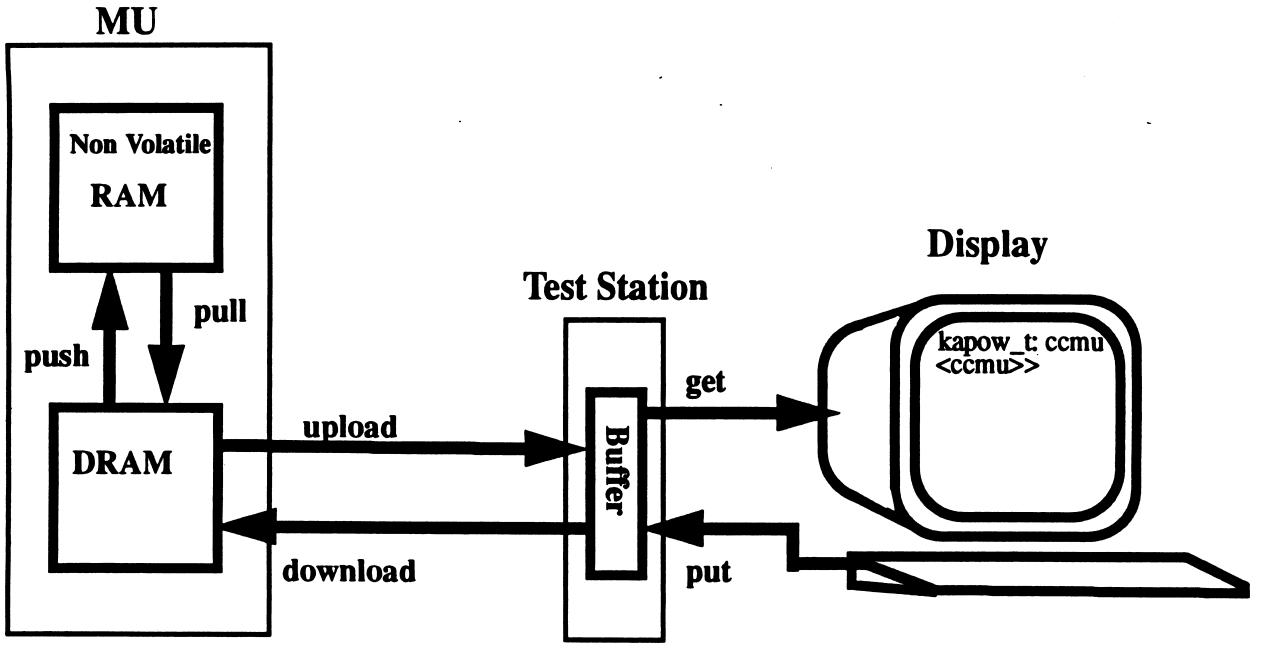
- **assign** <node #> <csb serial number> changes the address of the MU

  ## 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
- **query** [<node #> ...] [things] display all/ specified thing

14.4 The CCMU path diagram

Figure 14-2
CCMU diagram



task is now ——————————→ 588ØC1Øf to change it

bit 23 → PUloadOop → A

↑ comm/on →

↓

- put initwhat
- down → RAM
- push → NV RAM

14.5 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
```

14.6 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 MUERRENABLE INITLAND INITSLICE3
INITSLICE2 INITSLICE1 INITSLICE0 INITCXBAR DORESET PWRCLKCHECK GO

ccmu> put initwhat 0x5800cf0f
ccmu> down
ccmu> push
```

14.7 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.

14.7.1 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).

14.7.2 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

14.7.3 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 14.24 on page 163.
- 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 14.25 on page 164 for more details.

14.8 Xconfig All

XConfig is a GUI Hardware configuration utility

Figure 14-3
Xconfig Display

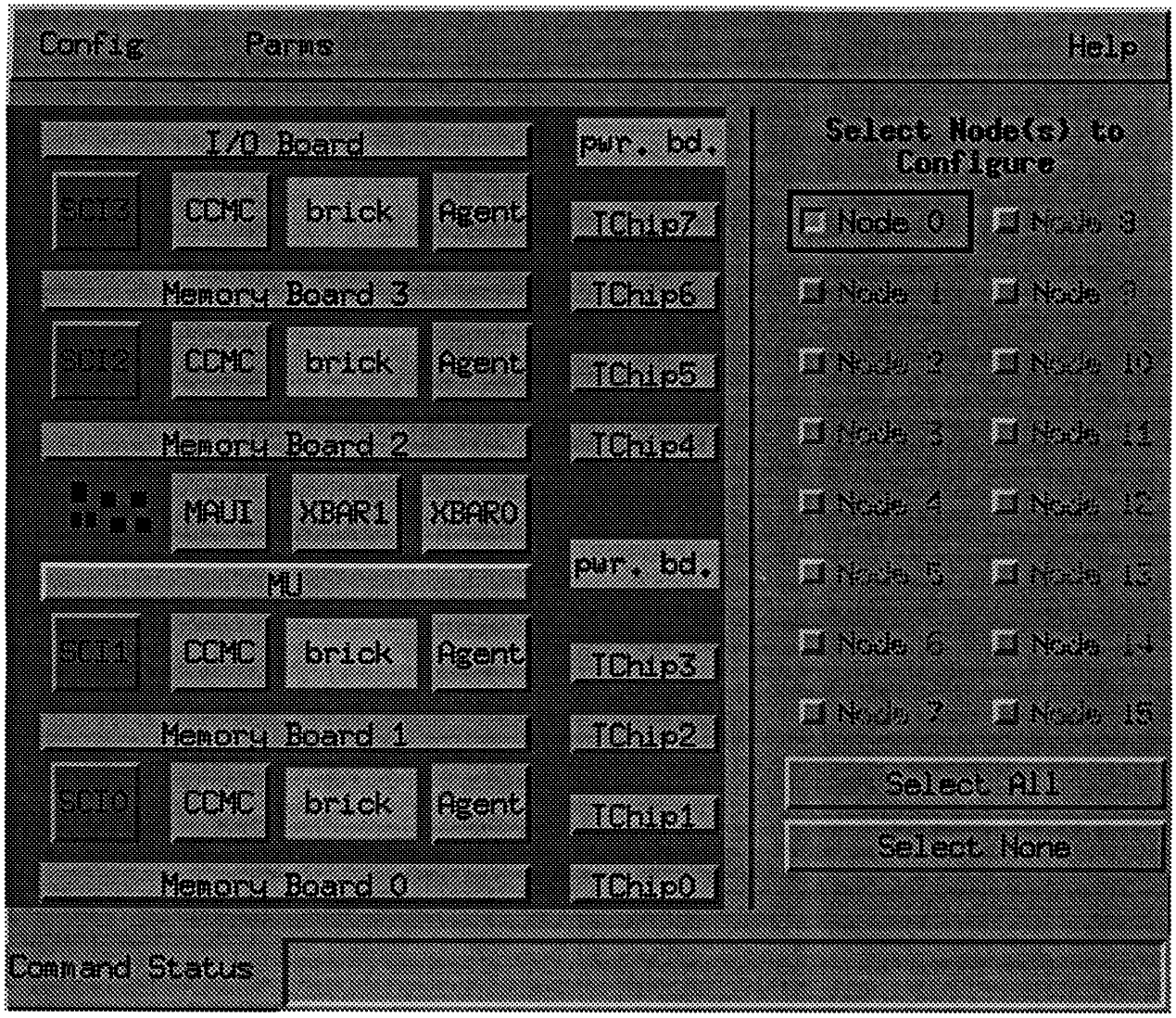


Figure 14-4
Xconfig Config Pop-up Menu

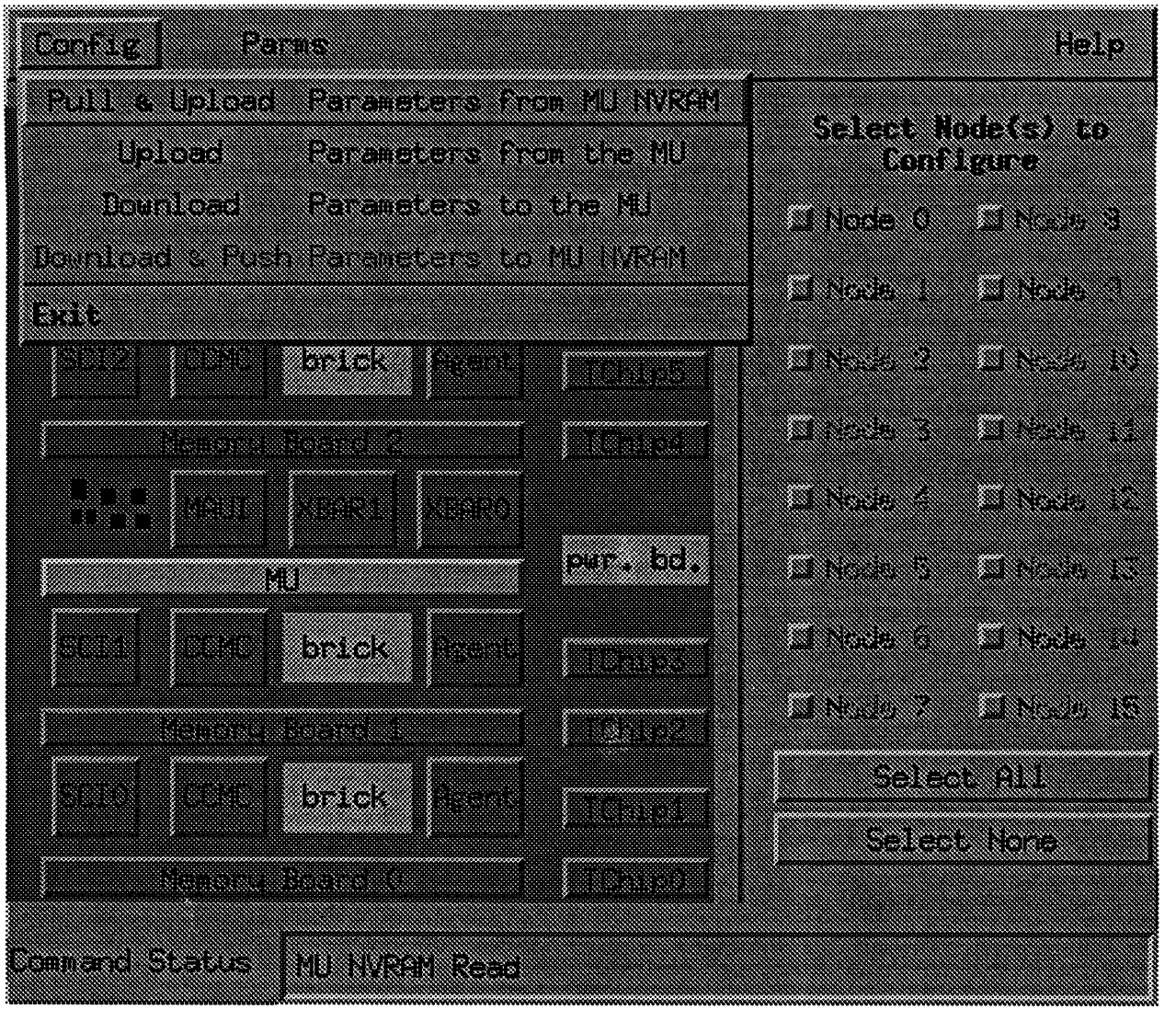


Figure 14-5
Xconfig Parms Pop-up Menu

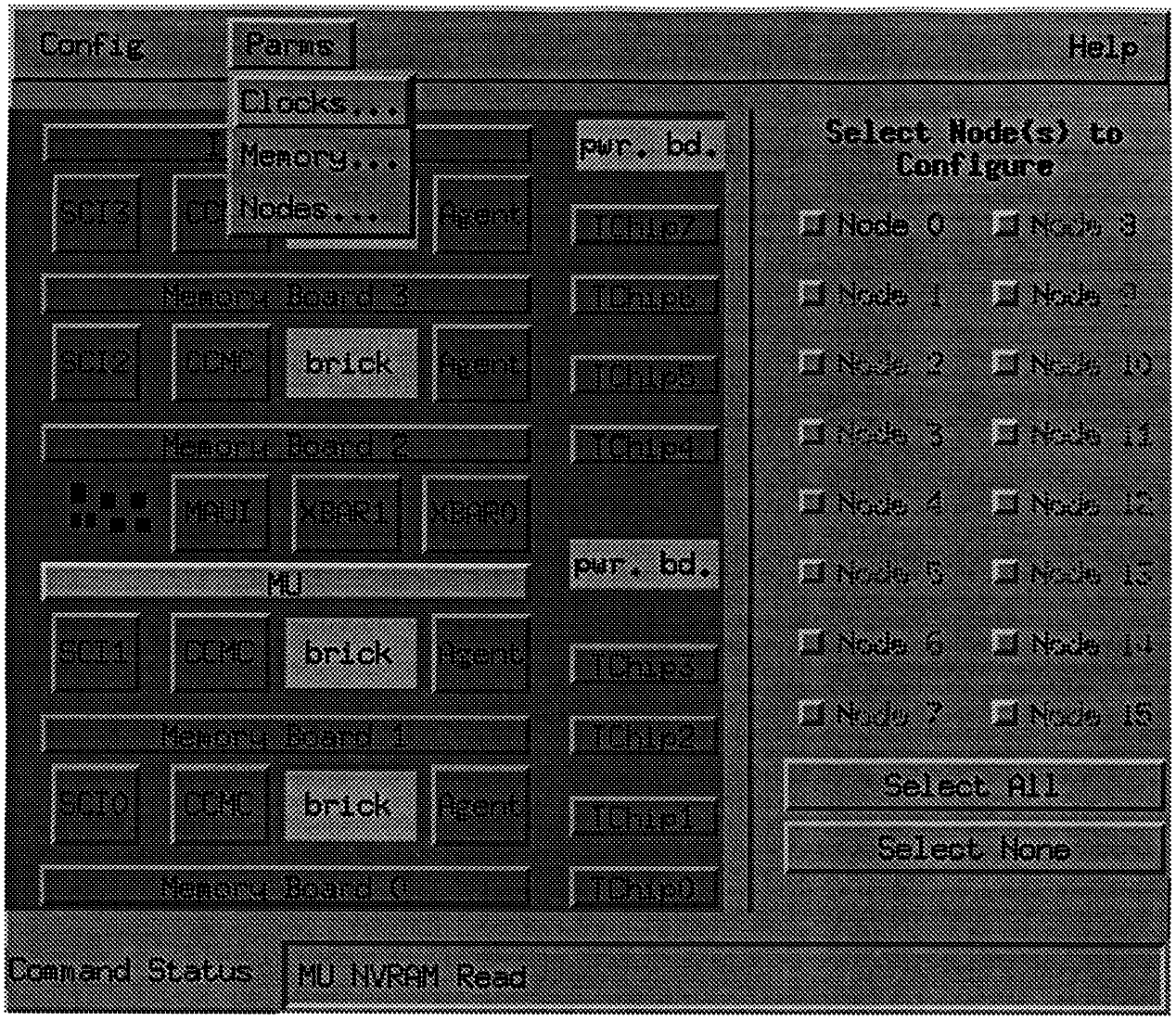


Figure 14-6
Xconfig Parms/Clocks Pop-up Menu

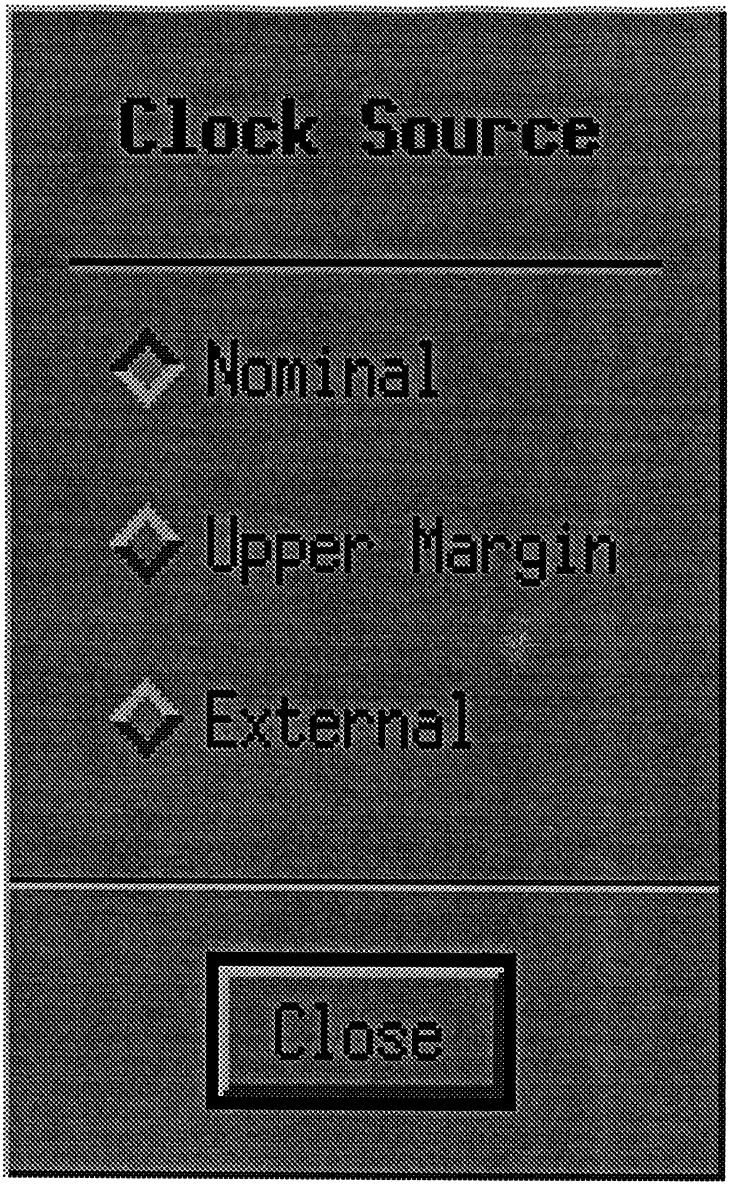


Figure 14-7
Xconfig Parms/Memory Options Pop-up Window

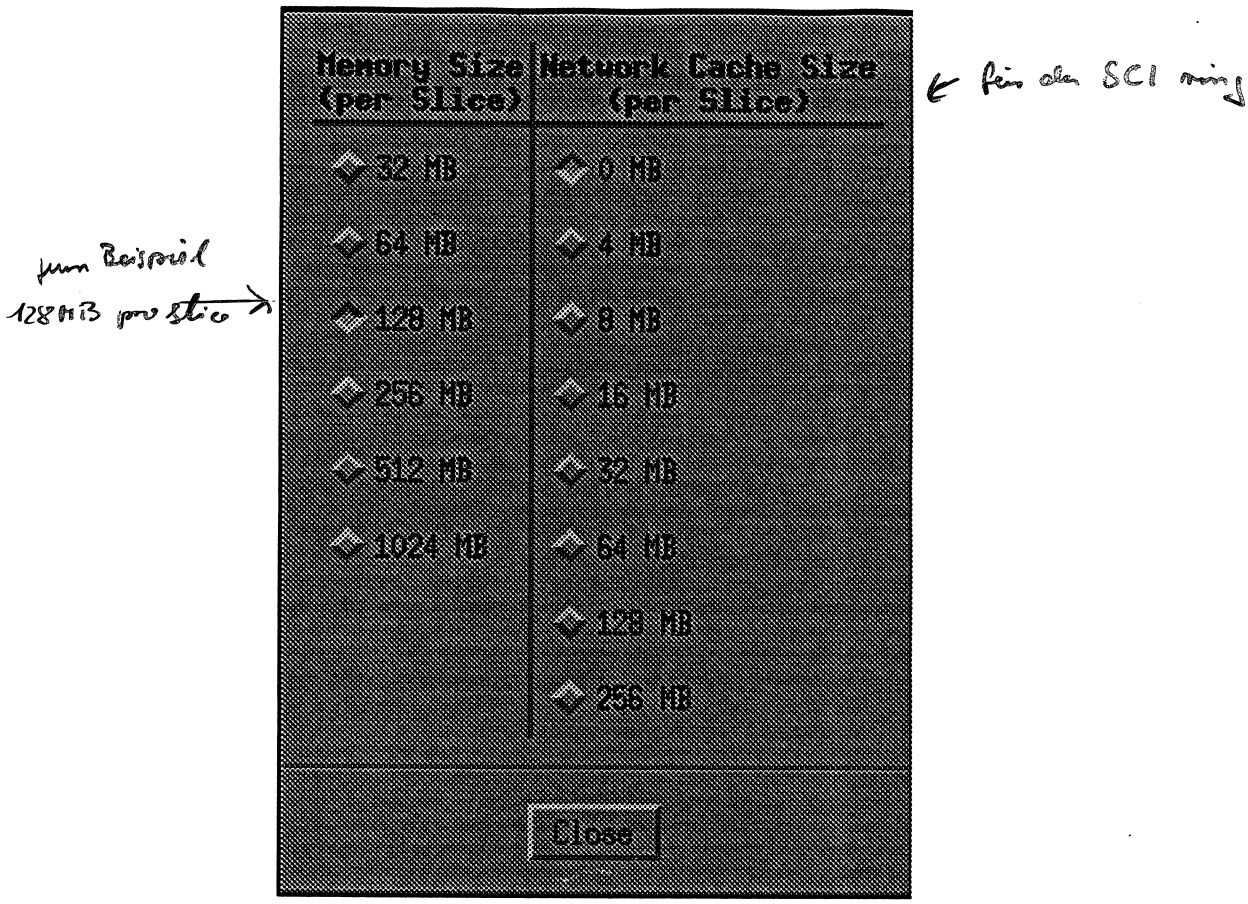


Figure 14-8
Xconfig Help Pop-up Window

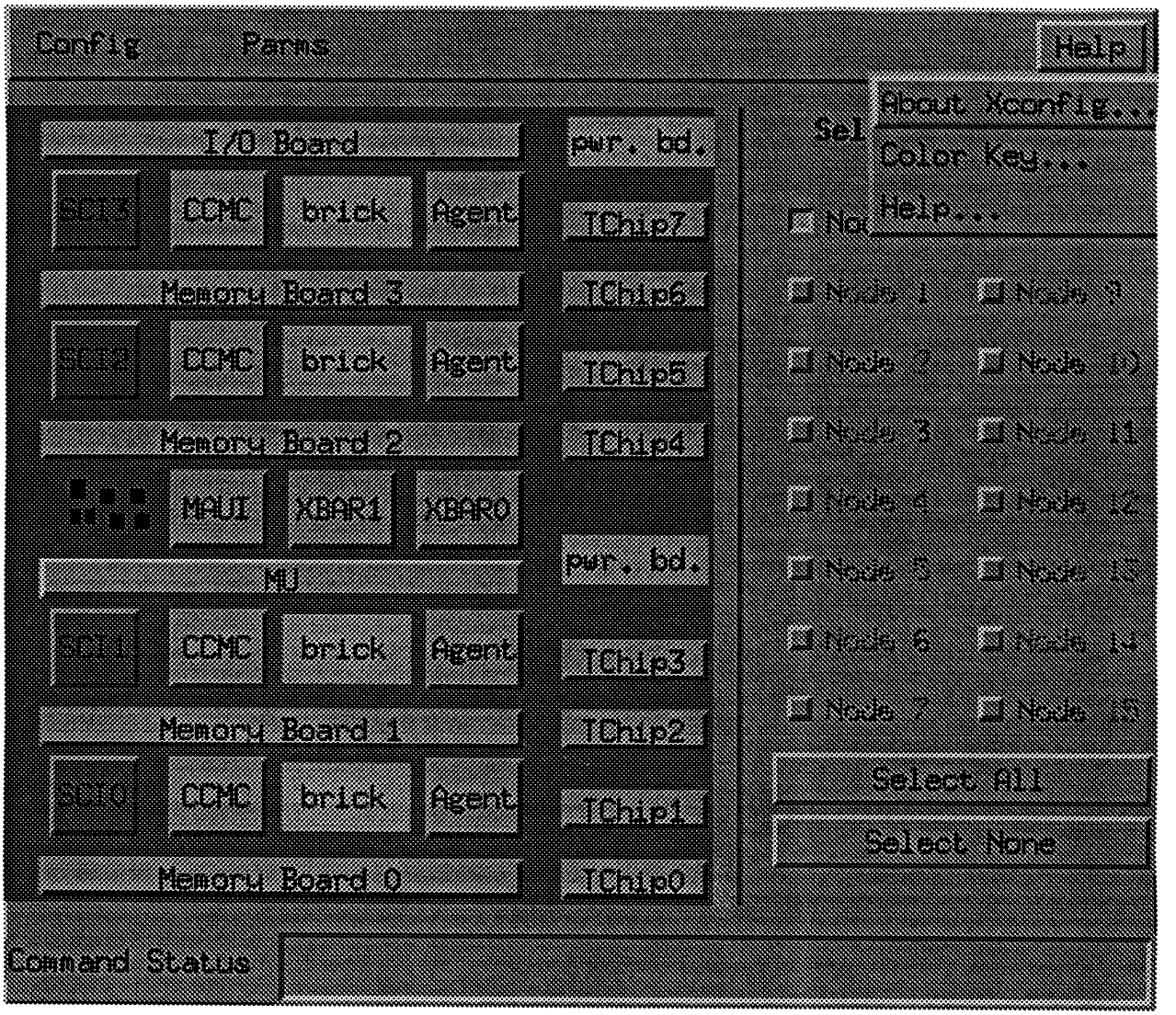
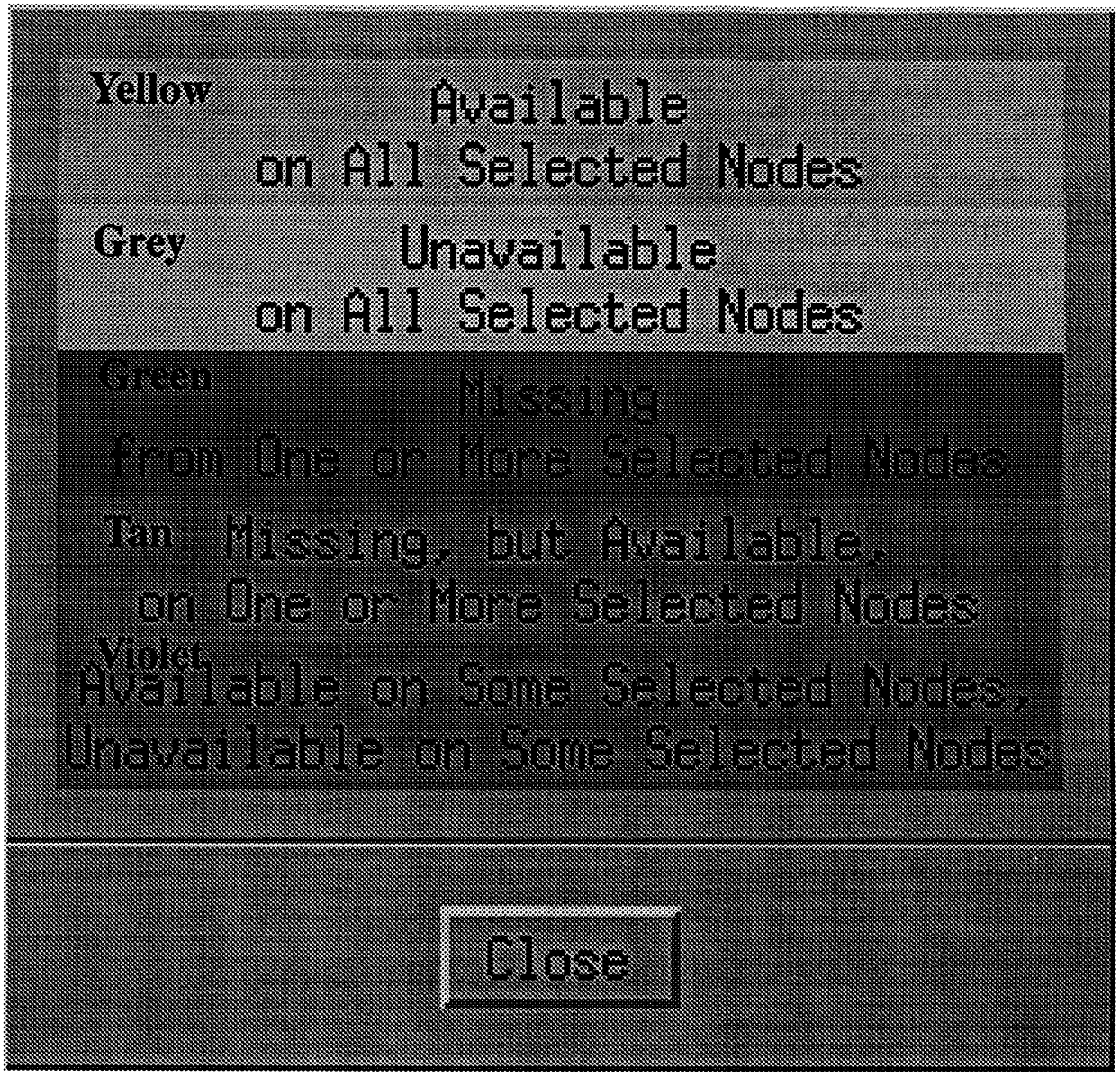


Figure 14-9
Xconfig Help/Color Key Pop-up Window



14.9 Cputest: T-CHIP Diagnostic

Provided by HP.

- EXAMPLE:

spp1_t% /spp/bin/cputest
subtest 1000
PASS/FAIL test

- 16 minutes per processor (runs on all processors on same node)
- nodes run independently

14.10 Cputest: T'-CHIP Diagnostic

- Runs in approximately 22 minutes no matter how many processors are under test
  - timeout is about 30 mins
  - failures are usually timeout
  - some data cache parity errors and HPMCs

14.11 CST - Convex Scan Test

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

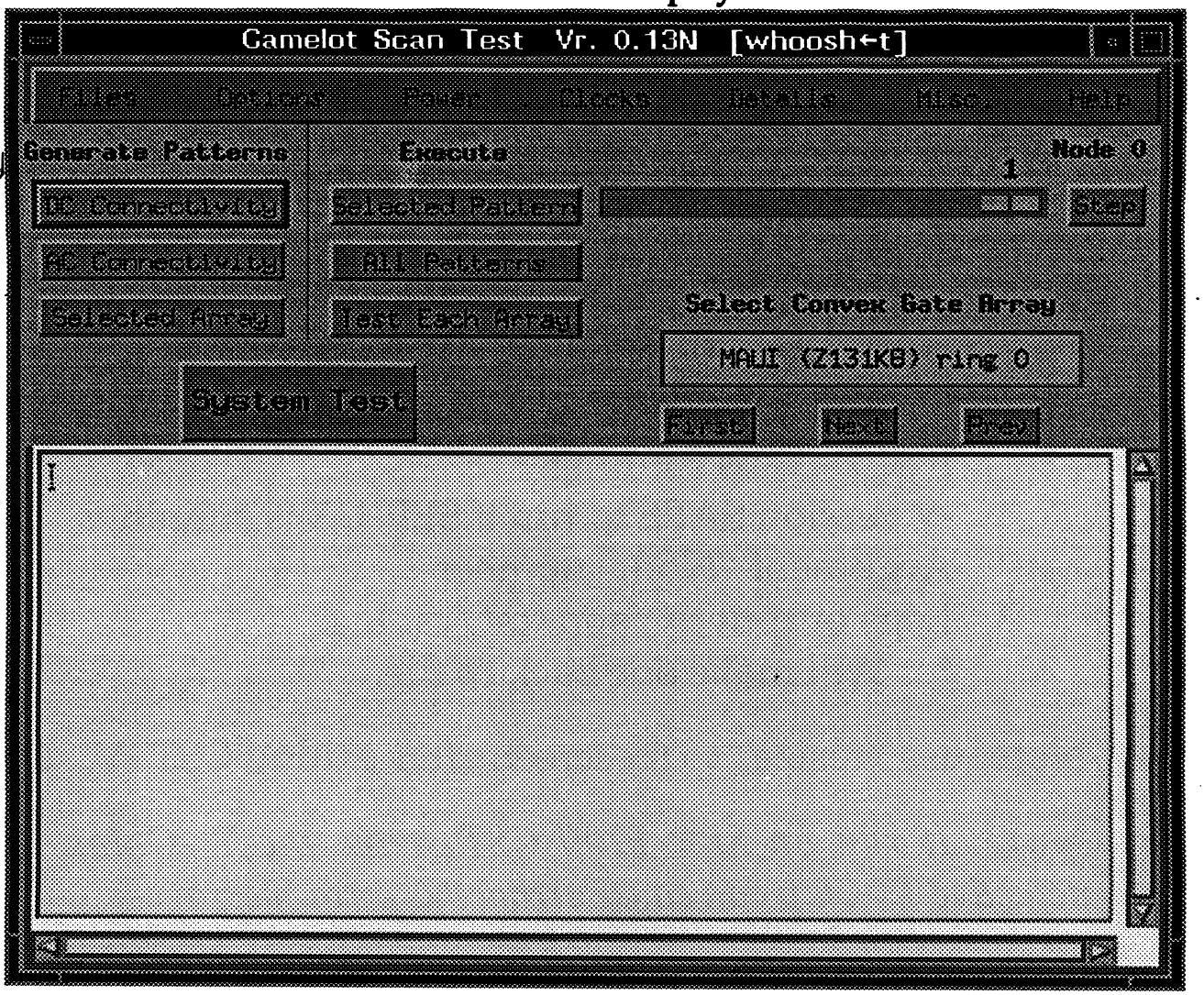
- Options:
  - **-V** dumb terminal mode.
  - **- N** demo mode (No Hardware calls) SAFE to run if SPP-UX is running.
  - **-v** version SAFE to run if SPP-UX is running.
  - Argument "Node_number".
  - Logs output to /spp/cst/cst.log.
    - Will make one backup file.

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 14-10
CST GUI Display



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

Figure 14-11
CST Scan Ring Partitioning

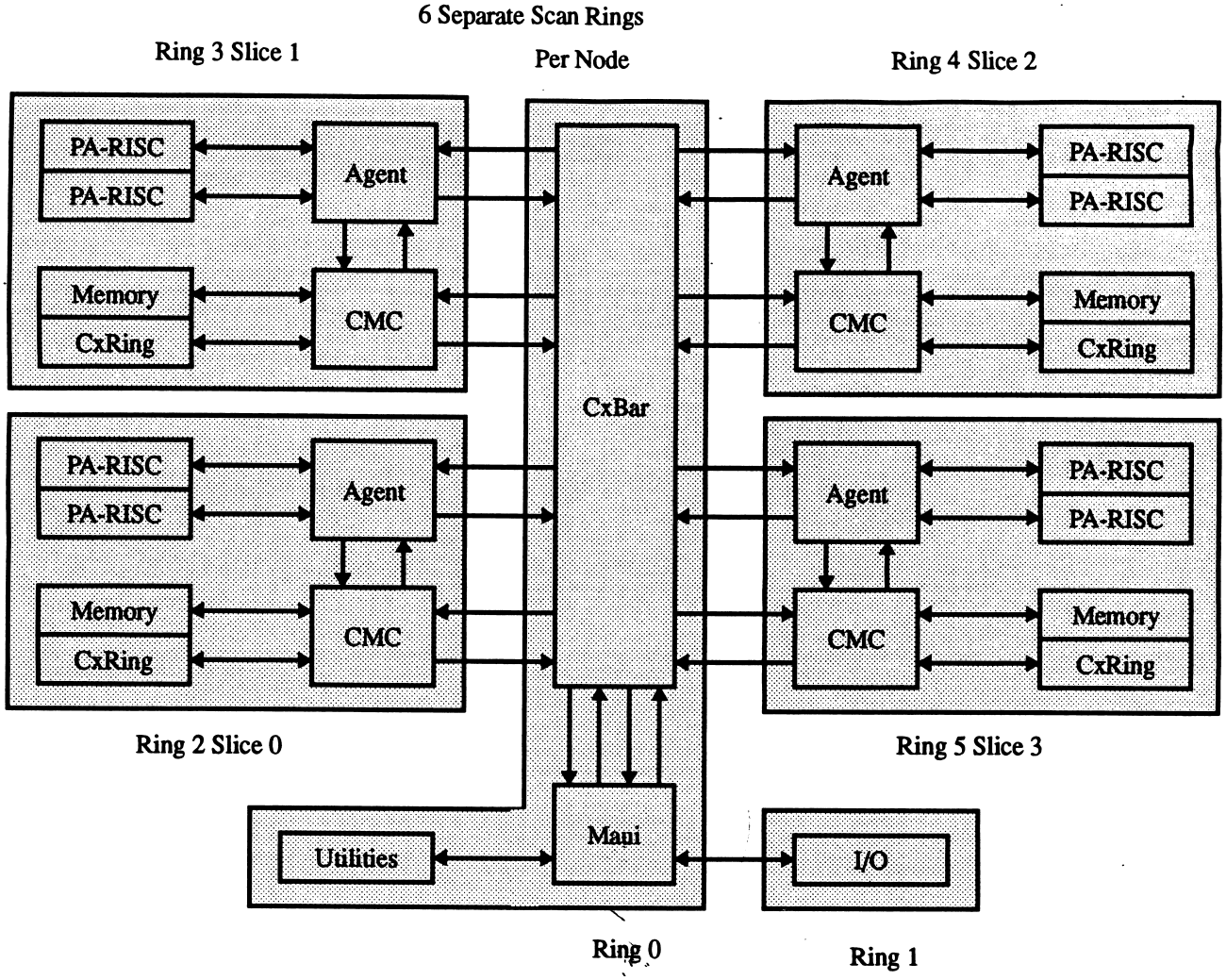


Figure 14-12
CST Files Menu

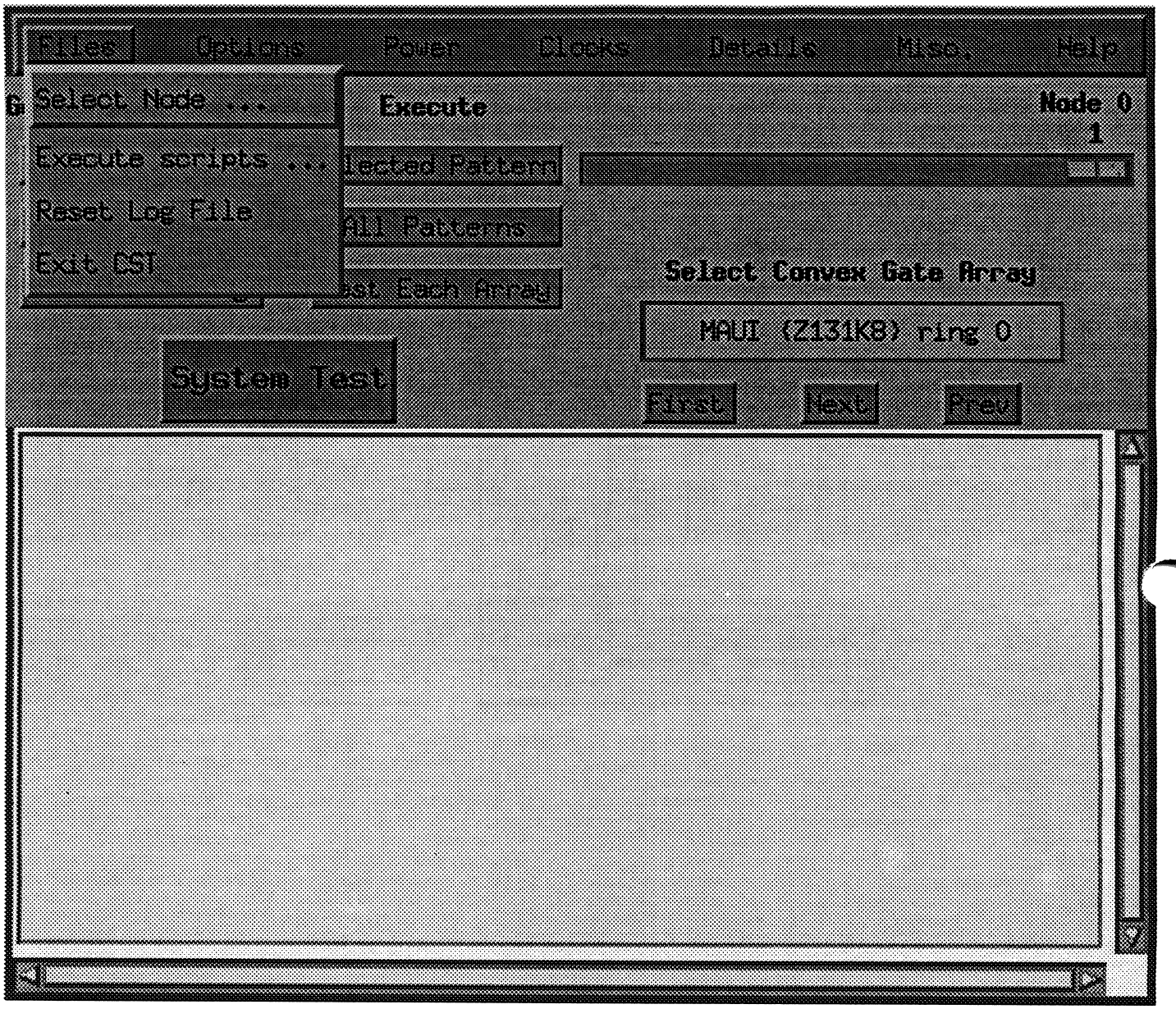


Figure 14-13
CST Options Menu

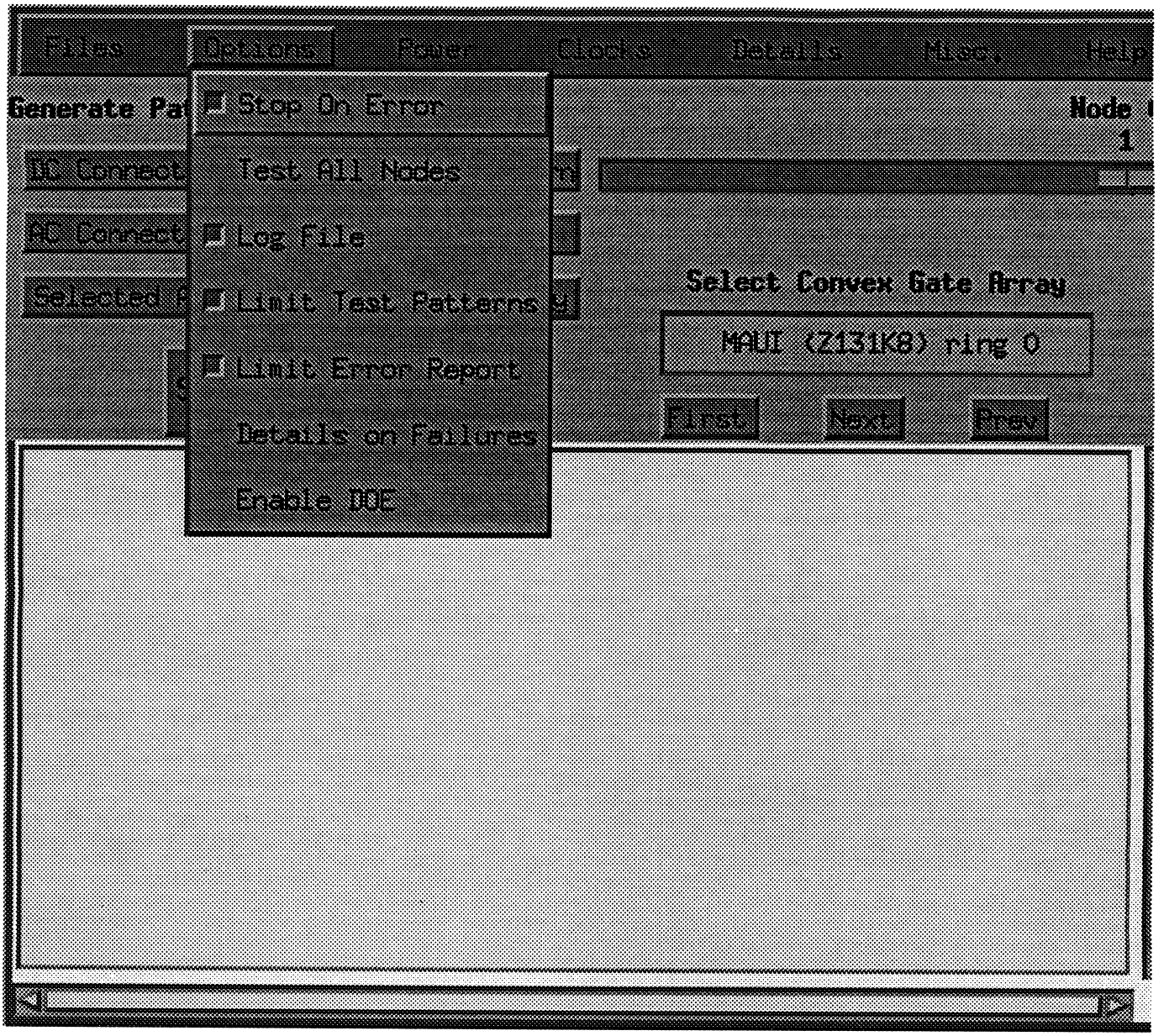


Figure 14-14
CST Power Menu

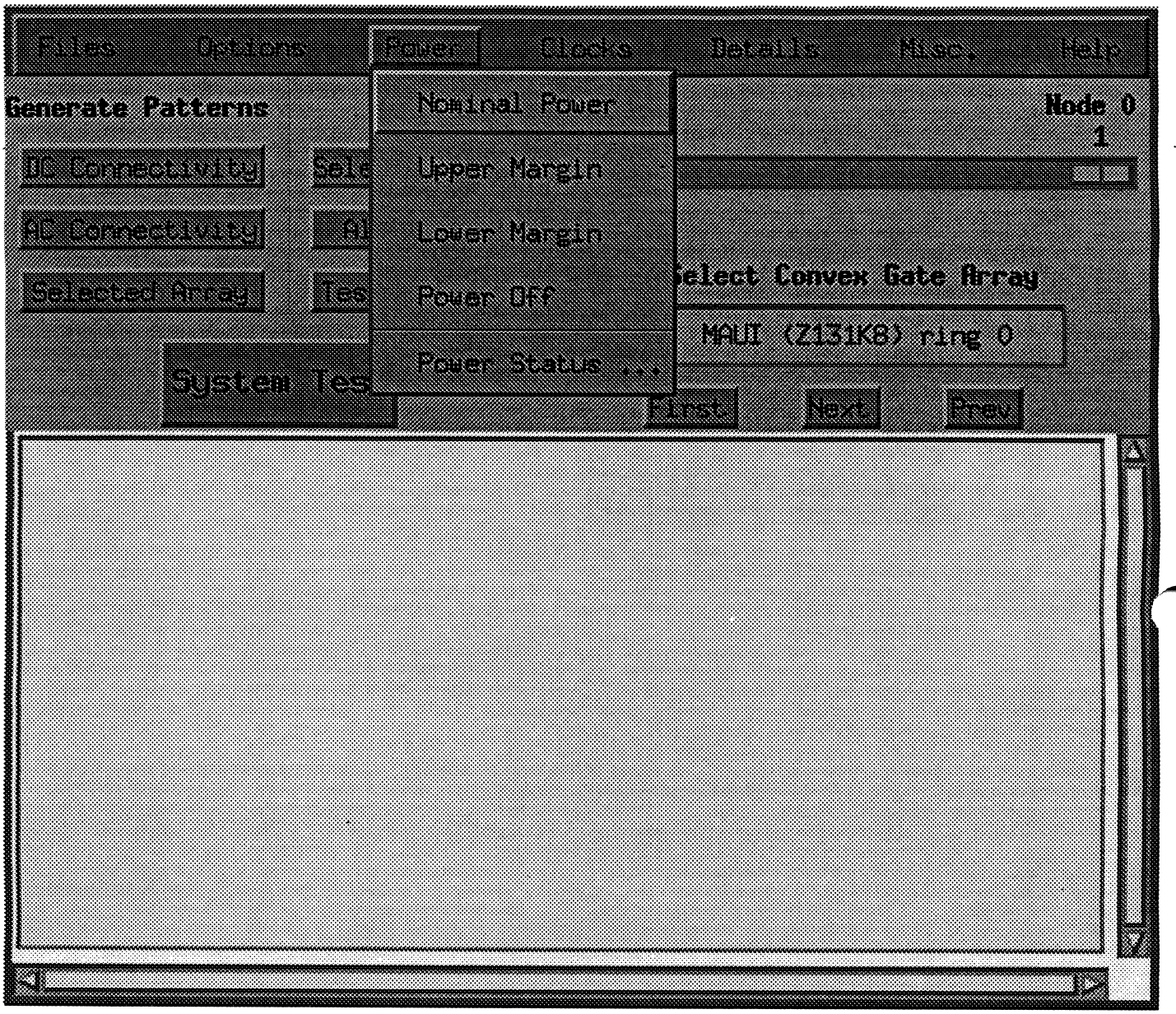


Figure 14-15
CST Clocks Menu

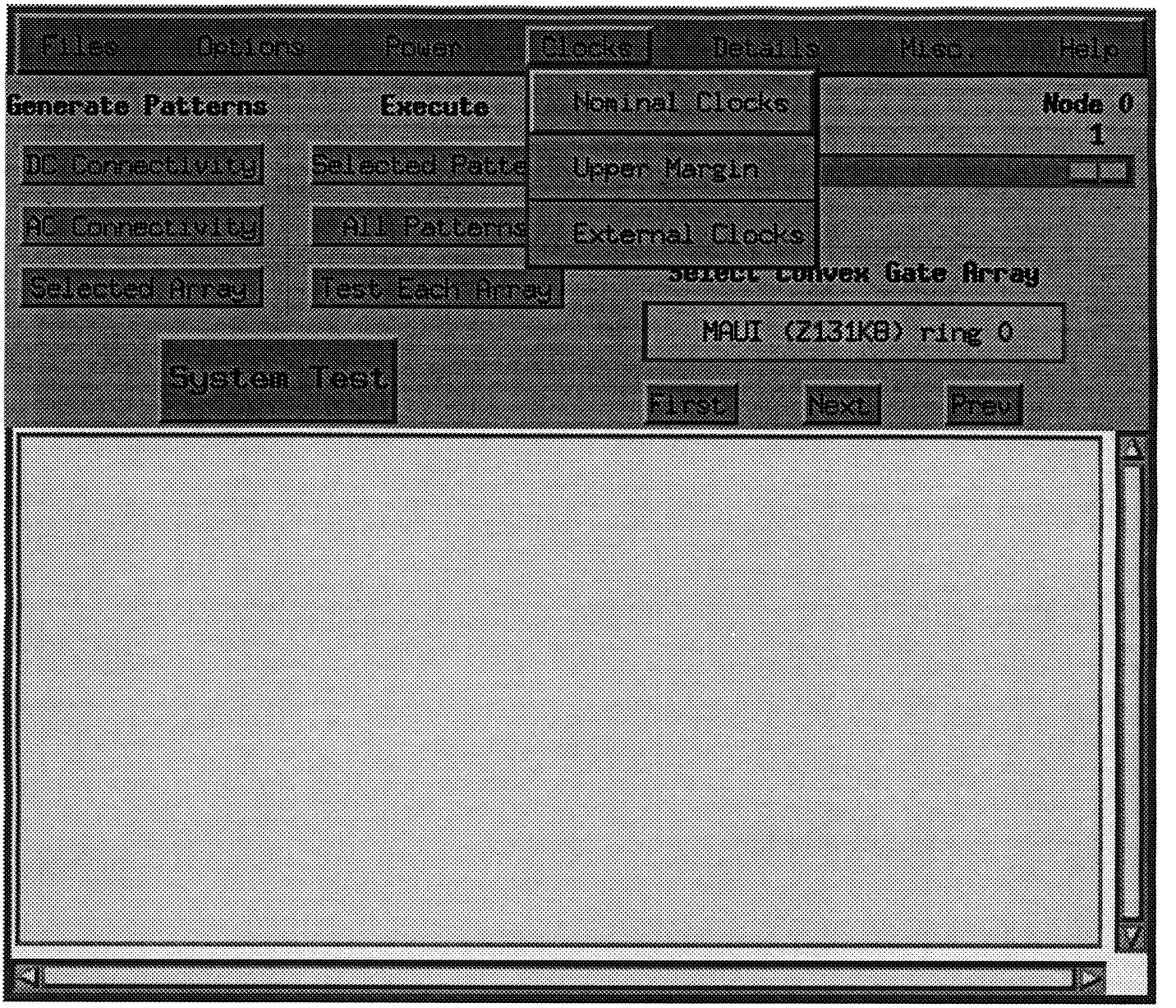


Figure 14-16
CST Details Menu

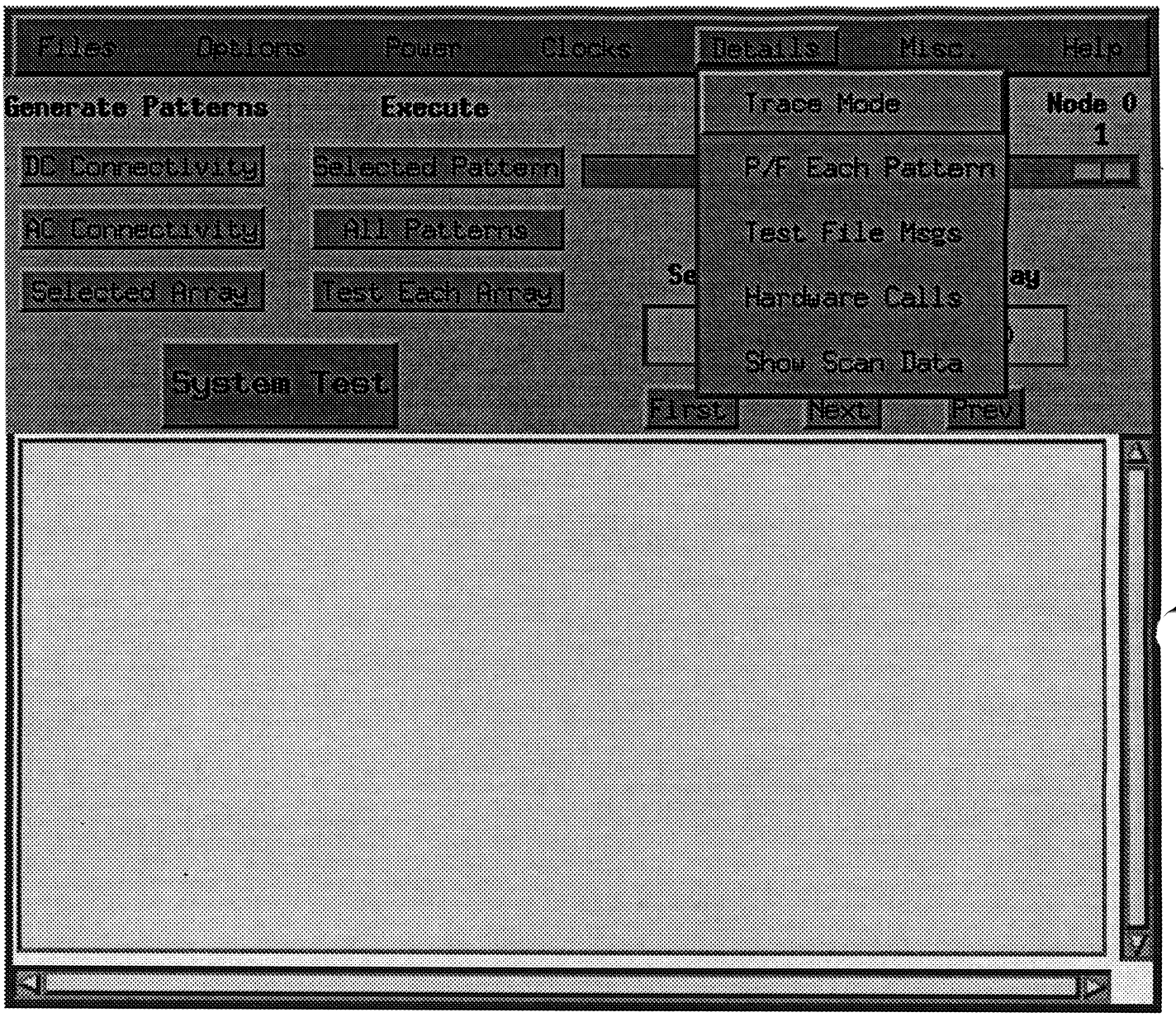


Figure 14-17
CST Misc. Menu

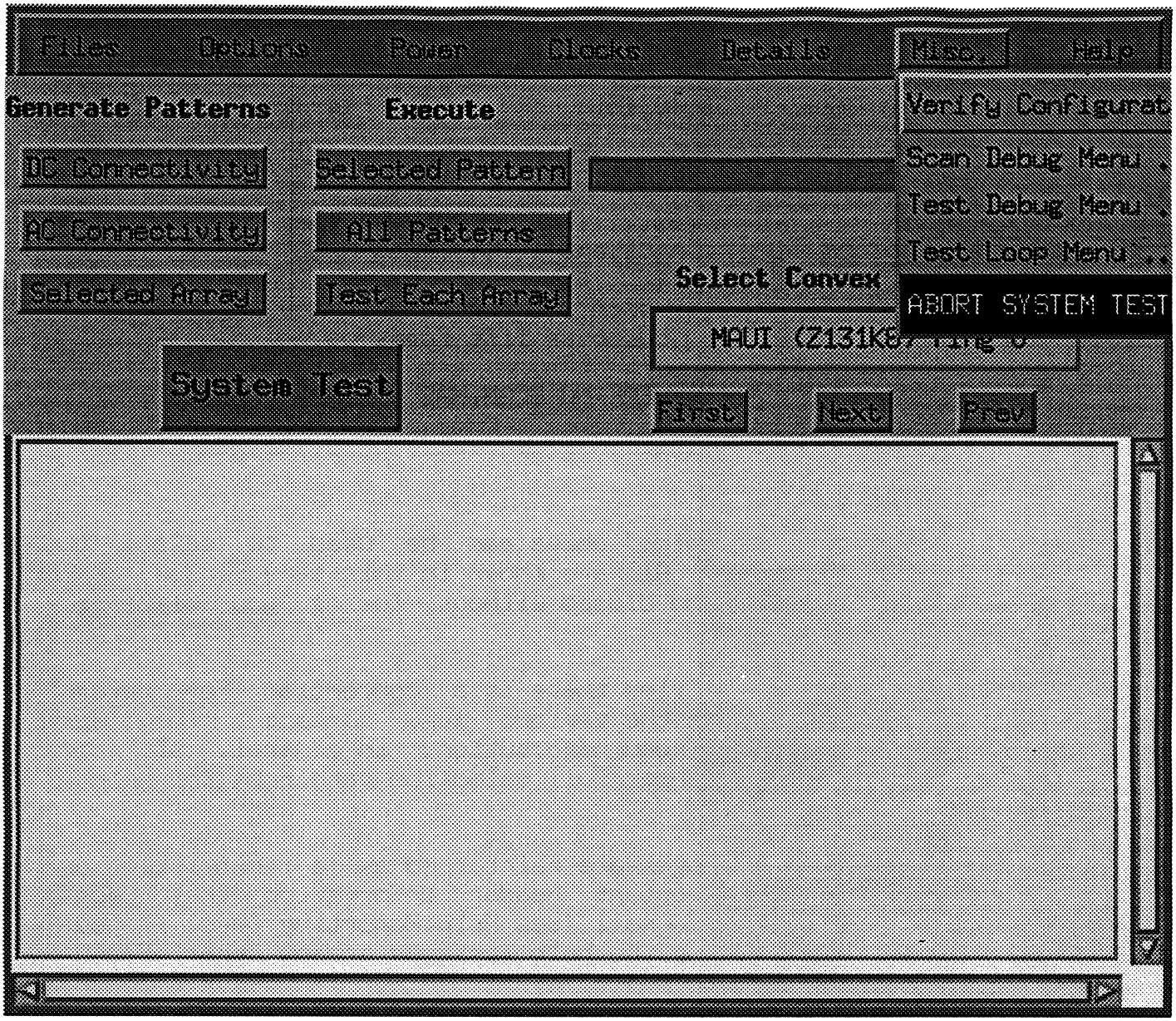
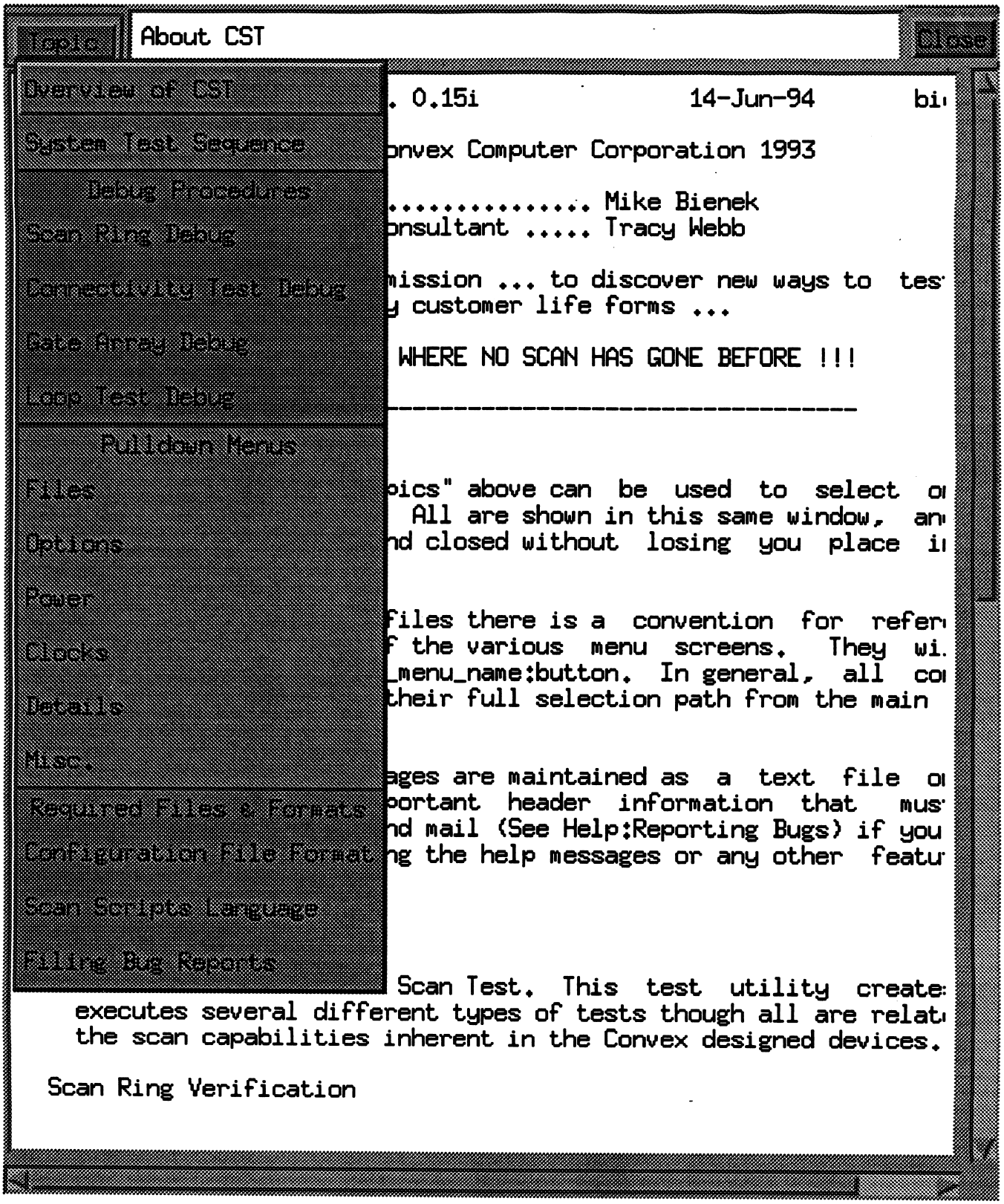


Figure 14-18
CST Help Screen



14.11.1 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
```

14.12 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.

14.12.1 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
```

Note: See Appendix •G on page 238, the STATUSMU value of 0x00000694 indicates the type of failure.

14.13 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.

- Example: /spp/data/event_log entry

warning bit

```
+++>  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 )
```

14.14 Event_logger MPP Test Station Event Logger

event_logger - receives messages from diagnostic utilities (through the event deamon) 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.

14.15 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 "Determining the Source of a Hard Error" on page 210 for a bit map list of the First Hard Error register.

14.16 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 landfill 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)

14.17 io_tc

- io_tc is an I/O diagnostic ran by the TCIPs, 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
```

```
        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
```

14.18 kill_by_name

Kills processes based on their names vs. process ID's

- Argument Req./Optional Description
  - 1 - required process name to kill
  - 2 - optional signal to send process (default is that of kill command)
  - 3 - optional pid not to kill, i.e. kill all processes that match the specified name except for the one matching this pid

14.19 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)

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.

14.20 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**

14.21 Reloading the Primary Loader

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

14.22 Reloading the Secondary Loader

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

14.23 New Versions of the load Commands

factory firmware →

- **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.

14.24 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**

14.25 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.

14.26 mp1_map - Address Decode Tool

Address to Hardware decode tool.

Memory configuration

DRAM Size(Mbit)	**= 4** (16MB)
Virt -> Phys	**= 0 1 2 3**
Banks/board	**= 2**
Rows per simm	**= 2**
Boards	**= 4**
Interleave	**= 4**
Physical address	**= 0x000000000**
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  >>Mmory 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
```

cpu - test
CST - test
sppmip - test

14.27 Node_info

A utility that quickly gives you basic information about the node.

Example:

```
elmo_d:/spp/bin$ node_info
Node  Serial                      UDP
 ID   Number     IP Address     Base Port      MU Firmware Version String
---- -------- ---------------- --------- ------------------------------------
  0   1999688  130.169.128.0       675   "CONVEX Version Info:   2.0.2"
(the primary loader version number is given in the 3.X release)
```

14.28 Pce_util - SPP Power, Clock & Environment Utility

NAME

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

SYNOPSIS

pce_util [-c {s[ource] <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

of[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.

14.28.1 pce_util output

```
scr1_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
```

```
scr1_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
```

14.29Sn_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

ctl

- **^e c .** - to exit either session
- **^e c f** - to regain control on the test station

14.30SPP 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**

14.31 Sppring CxRing and Distributed Memory Test

14.31.1 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.

14.31.2 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.

14.31.3 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).

14.31.4 Class 4 Non-Coherent Subtests

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

14.31.5 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.

14.31.6 Class 6 Intranode Coherency Subtests

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

14.31.7 Class 7 CxRing Internode Coherency Subtests.

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

14.31.8 Class 8 Message Subtest

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

14.31.9 Class 9 SCI Stress Subtests

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

14.31.10 Class 10 Coherency Stress Subtests

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

14.31.11 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.

14.31.12 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.

14.31.13 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.

14.32 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_MSIZE0 and CMC_NODE_MSIZE1 CSRs when the CMC arrays are initialized.

- Here's the current default set of diagnostic subtests, since version 0.0.58.124, nothing has changed.

```
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
```

```
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                        ..... DISABLED
          (broken subtest)
          3140  loopback subtest                           enabled
          3150  interrupt subtest                          enabled
          ????  ring integrity subtest              ..... DISABLED
          (requires gate arrays and power)

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)
```

14.33 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".

14.34io_hang

The script io_hang is used to help you determine if the i/o board is hung-up.

- io_hang example:

```
sppdsh$ /spp/scripts/io_hang

Default node is 0
Default ccmc is 0
ERROR: reading address 0xffed0000
Landfill unit 1 is hung
Reseting 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 0xe7eddffe
0x80001afc 0x00000000 0x00000000 0x00000000
0x80005e1e 0x80005e1f 0x80005e20 0x80005e21
0x80005e22 0x80005e23 0x80005e24 0x80005e25
0x80005e26 0x80005e27 0x80005e28 0x80005e29
0x80005e2a 0x80005e2b 0x80005e2c 0x80005e2d
```

14.35 dump_err

```
sppdsh$cd /spp/scripts/cpa
sppdsh$dump_err

Default node is 0
Default ccmc is 0
------------------------------------------------------------
 CPA 0
------------------------------------------------------------
 PROC 0
./dump_err[8]: hdb_display: not found
 PROC 1
./dump_err[10]: hdb_display: not found
 Internal

n0:a:internal:ag0:csri.csr_local.csrl_regs.r_hard_err_msw (4 bits) is 0x0 (0)
n0:a:internal:ag0:csri.csr_local.csrl_regs.r_hard_err_lsw (32 bits) is 0x0 (0)
n0:a:internal:ag0:csri.csr_local.csrl_regs.r_exc_cause0 (32 bits) is 0x0 (0)
n0:a:internal:ag0:csri.csr_local.csrl_regs.r_exc_cause1 (32 bits) is 0x0 (0)
n0:a:internal:ag0:csri.csr_local.csrl_regs.r_error_enable (11 bits) is 0x7cf
(1999)
./dump_err[17]: apq_d: not found
./dump_err[18]: port_d: not found
------------------------------------------------------------
 CPA 1
------------------------------------------------------------
 PROC 2
./dump_err[25]: hdb_display: not found
 PROC 3
./dump_err[27]: hdb_display: not found
 Internal
n0:b:internal:ag1:csri.csr_local.csrl_regs.r_hard_err_msw (4 bits) is 0x0 (0)
n0:b:internal:ag1:csri.csr_local.csrl_regs.r_hard_err_lsw (32 bits) is 0x0 (0)
n0:b:internal:ag1:csri.csr_local.csrl_regs.r_exc_cause0 (32 bits) is 0x0 (0)
n0:b:internal:ag1:csri.csr_local.csrl_regs.r_exc_cause1 (32 bits) is 0x0 (0)
n0:b:internal:ag1:csri.csr_local.csrl_regs.r_error_enable (11 bits) is 0x7cf
(1999)
./dump_err[34]: apq_d: not found
./dump_err[35]: port_d: not found
------------------------------------------------------------
 CPA 2
------------------------------------------------------------
 PROC 4
./dump_err[42]: hdb_display: not found
 PROC 5
./dump_err[44]: hdb_display: not found
 Internal
n0:c:internal:ag2:csri.csr_local.csrl_regs.r_exc_cause0 (32 bits) is 0x0 (0)
n0:c:internal:ag2:csri.csr_local.csrl_regs.r_exc_cause1 (32 bits) is 0x2000000
(33554432)
n0:c:internal:ag2:csri.csr_local.csrl_regs.r_error_enable (11 bits) is 0x7cf
```

```
(1999)
./dump_err[51]: apq_d: not found
./dump_err[52]: port_d: not found
------------------------------------------------------------------
 CPA 3
------------------------------------------------------------------
 PROC 6
./dump_err[59]: hdb_display: not found
 PROC 7
./dump_err[61]: hdb_display: not found
 Internal
n0:d:internal:ag3:csri.csr_local.csrl_regs.r_hard_err_msw (4 bits) is 0x0 (0)
n0:d:internal:ag3:csri.csr_local.csrl_regs.r_hard_err_lsw (32 bits) is 0x0 (0)
n0:d:internal:ag3:csri.csr_local.csrl_regs.r_exc_cause0 (32 bits) is 0x0 (0)
n0:d:internal:ag3:csri.csr_local.csrl_regs.r_exc_cause1 (32 bits) is 0x0 (0)
n0:d:internal:ag3:csri.csr_local.csrl_regs.r_error_enable (11 bits) is 0x7cf
(1999)
```

14.36 dump_all

/spp/scripts/cpa/dump_all does basically the same thing as a dump_err only on all specified nodes, and more - it dumps the entire scan ring.

Usage:

dump_all node#|node_name [node#2... | node_name]

- node# - are the actual node numbers e.g. 0 3 4
- node_name - is the name assigned to a specific group of nodes

14.37 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:  => MAU (Utilities d, x BAR (QdS))
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:  I/O
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 2:  Slice 0 bottom
ERROR: writing address 0xfff85010
Write from node 0 to 1.  Exp=5555aaaa, Act=5555aaaa, Diff=00000000
ERROR: writing address 0xfff89010
Write from node 0 to 2.  Exp=5555aaaa, Act=deadbeef, Diff=8bf81445  FAIL ***
ERROR: writing address 0xfff81010
Write from node 0 to 0.  Exp=5555aaaa, Act=deadbeef, Diff=8bf81445  FAIL ***

ERROR: writing address 0xfff89010
Write from node 1 to 2.  Exp=5555aaaa, Act=deadbeef, Diff=8bf81445  FAIL ***
ERROR: writing address 0xfff81010
Write from node 1 to 0.  Exp=5555aaaa, Act=deadbeef, Diff=8bf81445  FAIL ***
ERROR: writing address 0xfff85010
Write from node 1 to 1.  Exp=5555aaaa, Act=deadbeef, Diff=8bf81445  FAIL ***

ERROR: writing address 0xfff81010
Write from node 2 to 0.  Exp=5555aaaa, Act=deadbeef, Diff=8bf81445  FAIL ***
ERROR: writing address 0xfff85010
Write from node 2 to 1.  Exp=5555aaaa, Act=deadbeef, Diff=8bf81445  FAIL ***
ERROR: writing address 0xfff89010
Write from node 2 to 2.  Exp=5555aaaa, Act=deadbeef, Diff=8bf81445  FAIL ***


*** RING 3:  Slice 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

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 0xfffc9010
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:

RING 4 = Slice 2
RING 5 = Slice 3 TOP on chassis

14.38 verify_rings

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

Usage: verify_rings

SPP-UX 15

15.1 Boot Command

- On the Test Station
  - ccmu pull

or

  - do_reset
  - wait for the OBP to become active

- In the OBP window
  - [0:2] ok boot

15.2 Network Files

- /etc/hosts
- /etc/netlinkrc
- /etc/src.sh
- /etc/src.csh
- /etc/set_parms

15.3 Disk Administration Under SPP-UX

As seen during boot, it can be determined if a disk drive has already been mapped.

- 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 un mapped drive:

```
[62000001 001e3288] scsi disk: disk 0:0:5:0 not mapped;
```

15.4 Unmap 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 sd3**

15.5 MAP Disk to device

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

- Usage: MAP Disk sd node:ctlr:tgt:lun to device
- Example: Diskutil: **MAP Disk sd 0:0:4:0 to sd3**
  - maps SCSI ID=4 to sd3

15.6 PARTITION Disk

Using diskutil:

- DiskUtil: **select disk sd3**
- DiskUtil: **MAKe Partition a Size 1000M Offset 0**
- DiskUtil:**MAKe Partition b Size 1000M after at**

15.7 NEWFS

These are a few examples of how to enable a partition for use using the newfs command:

- newfs -v /dev/rdsk/sd3a scalios
- newfs -v /dev/rdsk/sd3b scalios
- cnx_newfs - for file systems over 2 GIG Bytes

15.8 SPP-OS FYI

15.8.1 Remote console commands

- Take a snap-shot of the OS console from another workstation
  - rlogin to the test station
  - **/spp/bin/sn_cnsl -S 1** (node 0 =1, 1=2, etc..)
  - **(cntl)ec.** to exit
- Take control of the OS console
  - rlogin to the test station
  - **/spp/bin/sn_cnsl -F 1**
  - **(cntl)ec.** to exit

15.8.2 Passwords

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

15.9 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 thee 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

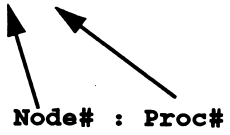
15.9.1 OBP Commands

- Often used OBP commands:
  - reset
  - printenv
  - setenv boot-device sd2a
  - setenv boot-directory /os
  - show-devs
  - devalias
  - nvalias sd0a /landmarc@0,ffec0000/sbus@f,fcffff00/Convex,afws@1,10000:narrow/sd@2,0:a
  - mkmap -n 0 /landmarc@0,ffed0000/sbus@f,fcffff00/CRES,cddi@1,400000
  - add-logical-units
  - boot

15.9.2 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
```



15.9.3 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
```

15.9.4 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

15.9.5 Important Boot Files

- On the test station
  - /spp/os/mach
  - /spp/os/server
  - /spp/os/tunables
  - /spp/os/sppboot
- On the SPP-1000
  - /os/mach
  - /os/server
  - /os/tunables

15.9.6 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)

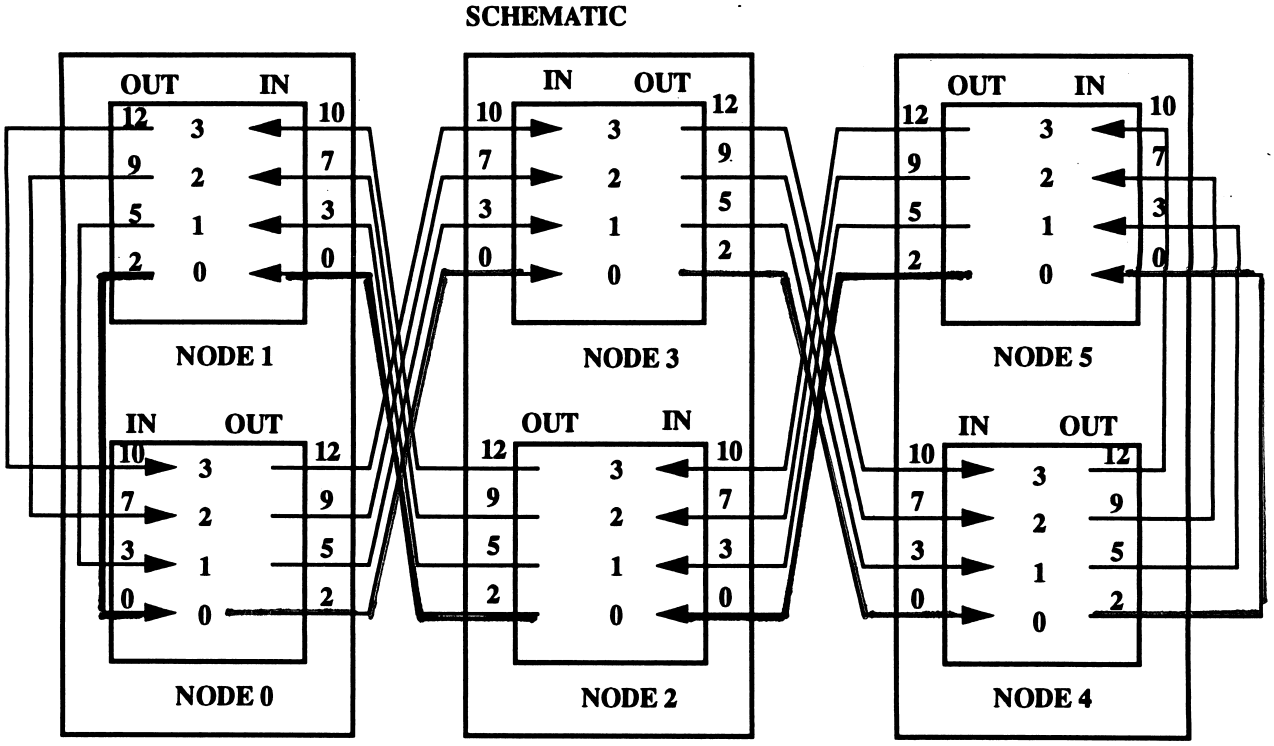
Multi-Node Operation & Setup 16

16.1 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.

16.2 SCI RING Connection

Figure 16-1
SCI XA Cabling Diagram



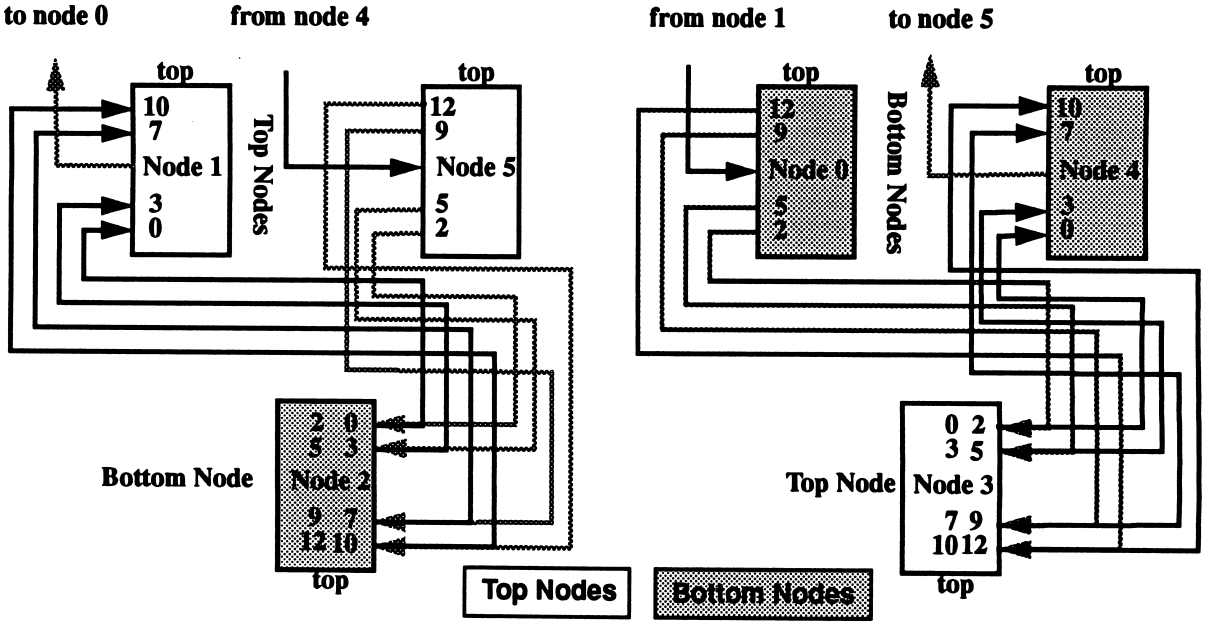


Figure 16-2
Node XA SCI Cabling Diagram

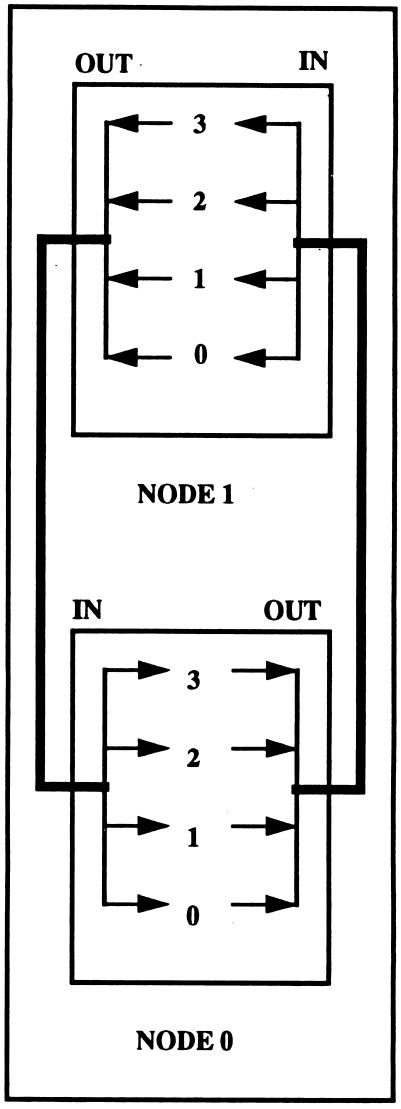
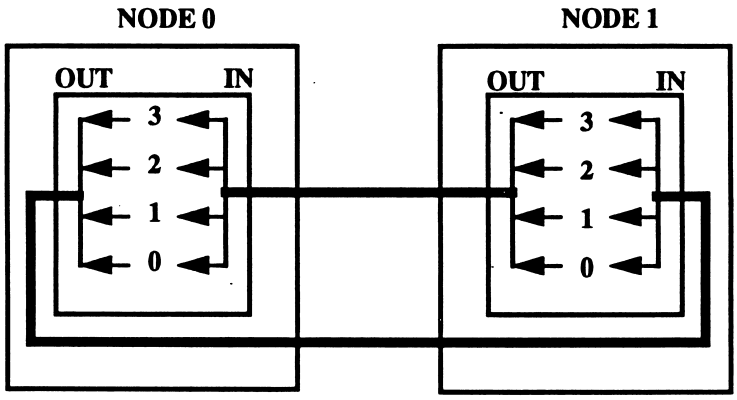


Figure 16-3
SCI CD Cabling Diagram



16.2.1 SCI diagnostics

- verify_rings
- verify_cables
- sppring -c(6,7,8,9,10) Cluster selection

16.2.2 Multi-cabinet connection & Power sequencing

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

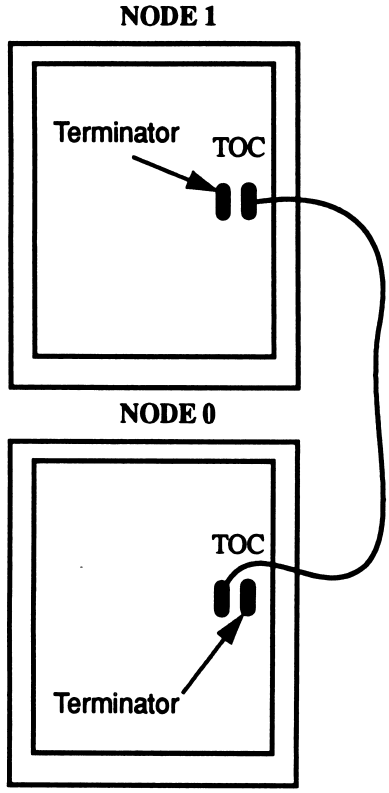
16.3 Current Multi-node HW Requirements

- CSB 4
- SCI2 Gatearrays (rev B)
- SCI Cables
- CCMC 2 (CMC dark)
- PB2 rev B
- CPA rev B

16.4 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 16-4
TOC cable connection diagram



Procedures and Trouble Shooting Aids 17

17.1 CD Installation Procedure

1) Open the CD 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) Close the node chassis.
6) If the DaRT 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 DaRT bus connection on the CSB.

- The power supply only is held in place with 2 screws.
- You can move the power supply with out disconnecting any cables.
- Remember to route the DaRT bus cable and the FDDI cable through the bulkhead on the back of the CD.

7) Replace the back cover and the power supply.

8) Verify the power switches on the node chassis and the peripheral chassis are in the ON position.

9) Verify the incoming power is correct and plug in the power cable.

10) Setup the test station.

11) Install the DaRT bus cable into the test station LAN port 0.

12) Power on the Node.

13) Wait until the front panel shows "Node init complete"

14) Power on the test station.

Note: The following steps are for a test station with 2 LAN ports, using LAN 0 for a private DaRT bus and LAN 1 for the public network.

15) Answer NO to the question asking you if you are ready to setup networking.

16) Once the test station is booted, login as root (root password is "serialbus").

17) Collect the following information:

- IP Address for the test station that will be connected to LAN 1.
- Router Address for the router to be used on the public network.
- Name of the test station.

18) cd /spp/scripts/inst

19) run ts.install (**./ts.install**)

20) Answer yes to the "use default IP addresses" question.

21) Answer all of the rest of the questions with the appropriate information you obtained at previous steps.

- Ignore any "can not create error messages"

22) Ignore the reboot the test station message if it exists.

23) Edit /etc/netlinkrc

24) search for the following statement in the 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
```

25) Remove the # sign in front of the next 12 lines in the file.

26) Save the changes and quit out of the editor.

27) Now Reboot the test station by typing /etc/reboot -n ,and cycling power on the test station when it tells you to do so.

28) Once the test station is rebooted login as sppuser (password = "spp user")

29) In the sppdsh window ping the node: **ping mu_0000**

If the ping is successful run these diagnostics in the following order:

30) cst 0

31) ccmu pull

32) sppring -s 1000-1050,2000-2020,2120,3000-3020, 3120,3220,4000-4010,5000-5010,6000-6050

33) ccmu pull

34) sysreset 3 0

35) io_tc

36) ccmu pull

37) sysreset 3 0 ——→ do_reset

38) type boot in the OBP window

17.2 Identifying When the IO Board is Hung

The io board is divided into 2 units. If you think one of the units has gotten hung and stopped working, there is a simple procedure you can follow.

1) Start up sppdsh.

2) Make sure the cxbar 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**"

Note: In the lab, we have the disks connected to unit 0.

- 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.
- For a fix, you can either run **/spp/bin/do_reset** or **/spp/bin/lm_reset**. lm_reset just toggles the reset lines to the io board.

Note: If you do reset the io board, you may want to cycle

power on the scsi drives. If the io board got hung during a scsi operation, the drives might be "hung" as well.

17.3 Autoconfig network on boot

The first time SPP-OS is booted on install you will be prompted for network information. This information will be used to autoconfigure 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

- If you plan to have two Networks connected to the test station answer no to the setup network question during boot.

1) Login as root
2) cd /spp/scripts/inst
3) ./ts.install
4) do not reboot now
5) Edit /etc/netlinkrc (see Section 17.4.1 "/etc/netlinkrc")
6) reboot the test station

17.4 Initial install network setup

17.4.1 /etc/netlinkrc

- If the test station has 2 LAN's do the following:
  - Look for the following lines:

```
# 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.

17.5 Determining the Source of a Hard Error

When a hard error occurs in the node, the MU reports the error on the LCD and sends an event to the test station.

- (This only happens if the MUERRENABLE bit is set in the INITWHAT config parameter.)
- To determine the source of the hard error, you must read some registers on the MU.
- These registers are:
  - 0xf0c04050 - hard error source
    - - tells you which hard errors occurred.
  - 0xf0c04058 - hard error mask
    - - tells you which hard errors are enabled.
  - 0xf0c0405c - first hard error
    - - tells you which hard error occurred first.
- The bits in these registers are defined as:

15-13	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

welcher agent

welcher ccmc

17.6 SPP-UX Hang and Panics Procedure

SPP-UX Hangs and Panics

- If the SPP hangs or panics during SPP-UX, do the following to capture pertinent data for the SPP-UX folks.
  - This has been (will be) replaced with "crashdump" utility. See the SPP-UX System Administration Guide for details on how to run it.

1) 1. In a test station sppuser "sppdsh" window, push <CTRL> and the left mouse button. Select "Log To File". This will direct all output (for this window) to the file / users/sppuser/XtermLog.XXXX. Where XXXX = pid of the xterm process.

2) 2. Check for hard errors. "mget 0xf0c0405c" If returned data is zero, no hard error has occurred, go to step 3. If returned data is non-zero, a hard error has occurred, check event_log for hard error info. Go to step 12.

3) 3. Check if we can access memory, thru the agent and xbar. "mget 0xffe80000" If this returns data, the memory is not hung, continue with the next step. If this returns an error message, the memory is hung, go to step 12.

4) 4. Check if the landfill board is hung. "io_hang" If the

landfill is hung, go to step 12. If the landfill is not hung, then SPP-UX is probably hung, go to step 5.

5) Startup gdb for a processor backtrace.

```
cd /spp/os
gdb mach_kernel
(gdb) target camelot [teststation hostname] 0
(gdb) attach 0      (Note: always attatch 0 for mach kernal)
(gdb 0:1)
#    ^ ^
#    ^ This is the cpu #
#    ^ This is the node #
# be aware that when you enter gdb you will not always get the
# same cpu.
```

6) Get a processor backtrace.

```
(gdb 0:1) backtrace
```

7) Dump the processor lock stack.

```
(gdb 0:1) print _processor_lock_stack [#]    (Where # = processor #)
                                              (Brackets are required)
```

8) Dump the cpu registers.

```
(gdb 0:1) info reg
(gdb 0:1) cs #              (Where # = next processor)
#
# Repeat steps 6-8 for each processor installed. #
```

9) (gdb 0:1) detach

10) Exit gdb (gdb 0:1) quit

11) Startup gdb for a server backtrace.

```
cd /spp/os
gdb server.exe
(gdb) target camelot [teststation hostname] 0
(gdb) source /spp/os/gdb_scripts/taskbt
(gdb) attach 3
(gdb) info threads (Get count of server threads)
(gdb) taskbt # (Where # is the number of the server thread
                to be backtraced. Backtrace each thread)
(gdb) detach
(gdb) quit
```

12) cd /spp/scripts/cpa; dump_err

13) Push <CTRL> and the left mouse button. Select "Log To File". This will cancel output logging. 1

14) Move the XtermLog.XXXX file to the XtermLog.ddmmyyhhmm.

15) cd /spp/os

16) sppboot or wait for further instructions.

17.7 Hard Error Debug (Agent hard error check)

On the systems that appeared hung, landfill board is not hung, and you were not able to attach to the kernel via gdb, it may be that you had a hard error which caused the system to stop.

- To check that you have a hard_error do a get of the register 0xf0c0405c.
  - This is the first hard_error register.
  - If there are bits set in this register, then there is a hard error.

- If there is a hard_error set check the file /spp/data/event_log to see what the hard_logger has found.

- In these cases after you have completed the dump_err script, check for the following in the output.

- CPA -

```
n0:a:internal:ag0:csri.csr_local.csrl_regs.r_hard_err_lsw (32 bits) is 0x400002
```

  - If the r_hard_err_lsw for any of the agents (:ag?:) in the above scan names are set to **0x400002** - this is an agent hard error.

The next time this happens do the following.

- Use ccmu to change the parameters list below to the values shown.
  - RUNWHAT 0xf30faaff (old value 0xf30f55ff)
  - AVAIL_HW 0x800f1f55 (old value 0x800f1faa)

- Be sure to do a down, and then a push when you are done modifying these inside ccmu to make sure they are saved.

- Then reboot the system, by doing a do_reset 3, and then the sppboot.

This will effectively switch processors the machine is going to run on without changing any of the installed hardware.

Appendix •A Class Documentation list

- **Student Handout**
- **Trouble Shooting Hints**
- **SPP Ring 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**
  Partnumber 710-029430-000
- **SPP-UX System Administration Guide**
  Partnumber 710-029330-000

Appendix •B Tools

- **315-000205-500 INSTALLATION ROD, ASIC 2 ea.**
- **9/16" socket. To put the feet of the towers down**
- **1/8" socket. To install SCSI bus cables at the I/O chassis**
- **Long, magnitized #2 Phillips screw driver.**
- **Short (stubby) #2 Phillips screw driver.**
- **1/4" tork socket wrench 5"lbs setting**

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

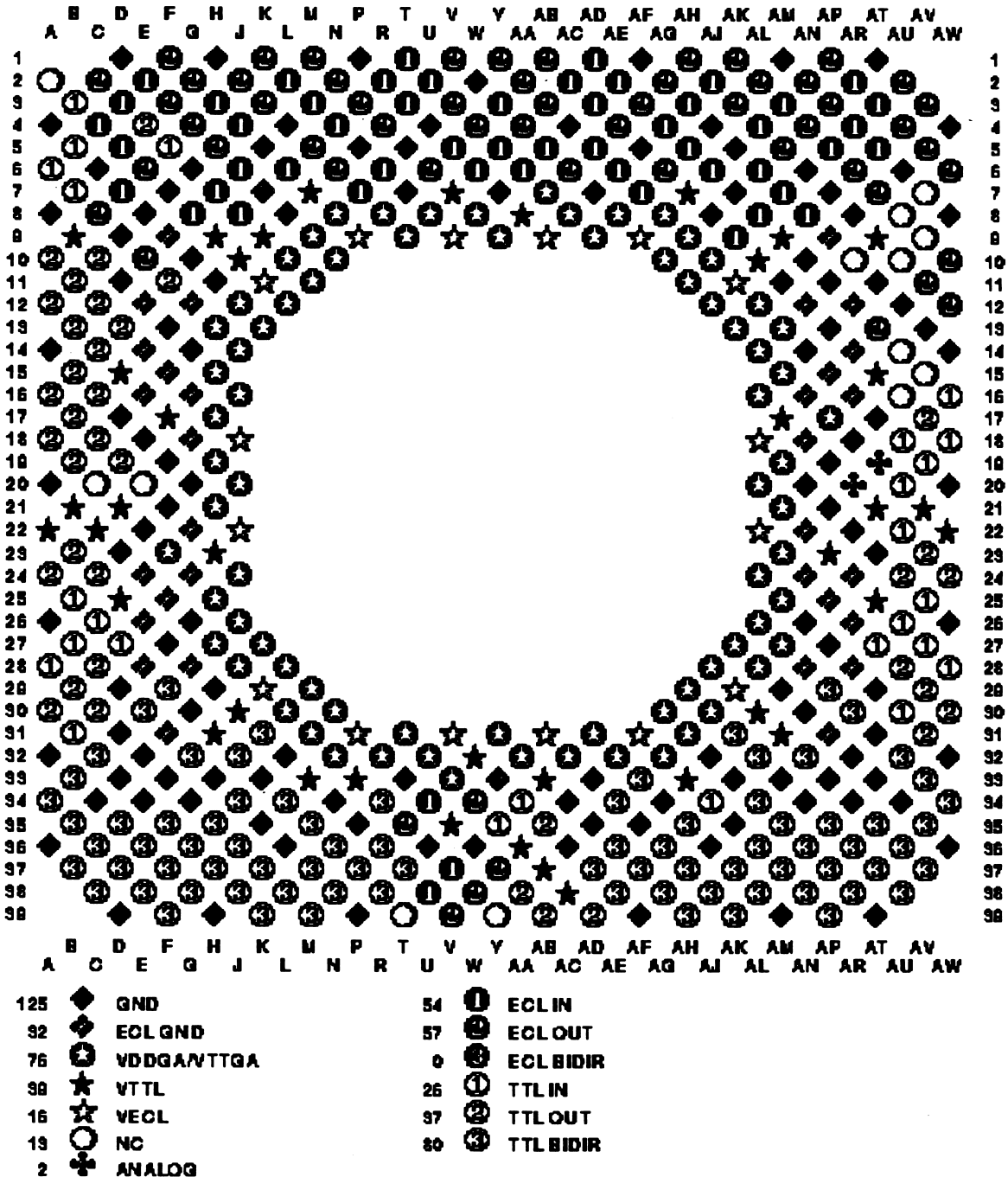
253-000124-001 FUSE, 3A 250V, PC-TRON
253-000124-003 FUSE, 5A 250V, PC-TRON
253-000128-001 FUSE, 10A/250V .25X1.25 FAST CER

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

Appendix •D Gatearray pin-out

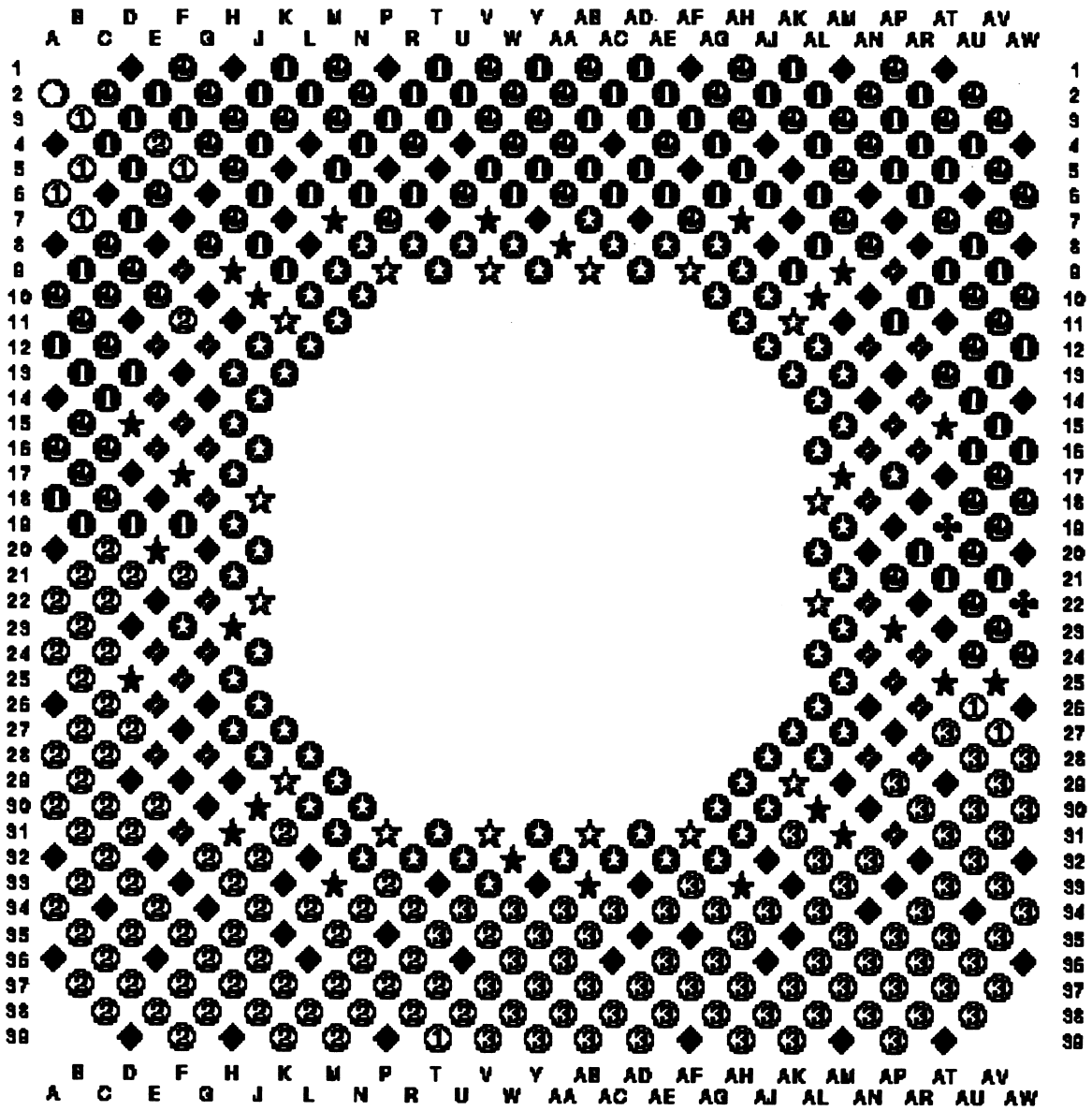
1 Agent

LGA557 PIN MAP FOR AGENT [probe-side view]



2 CCMC2

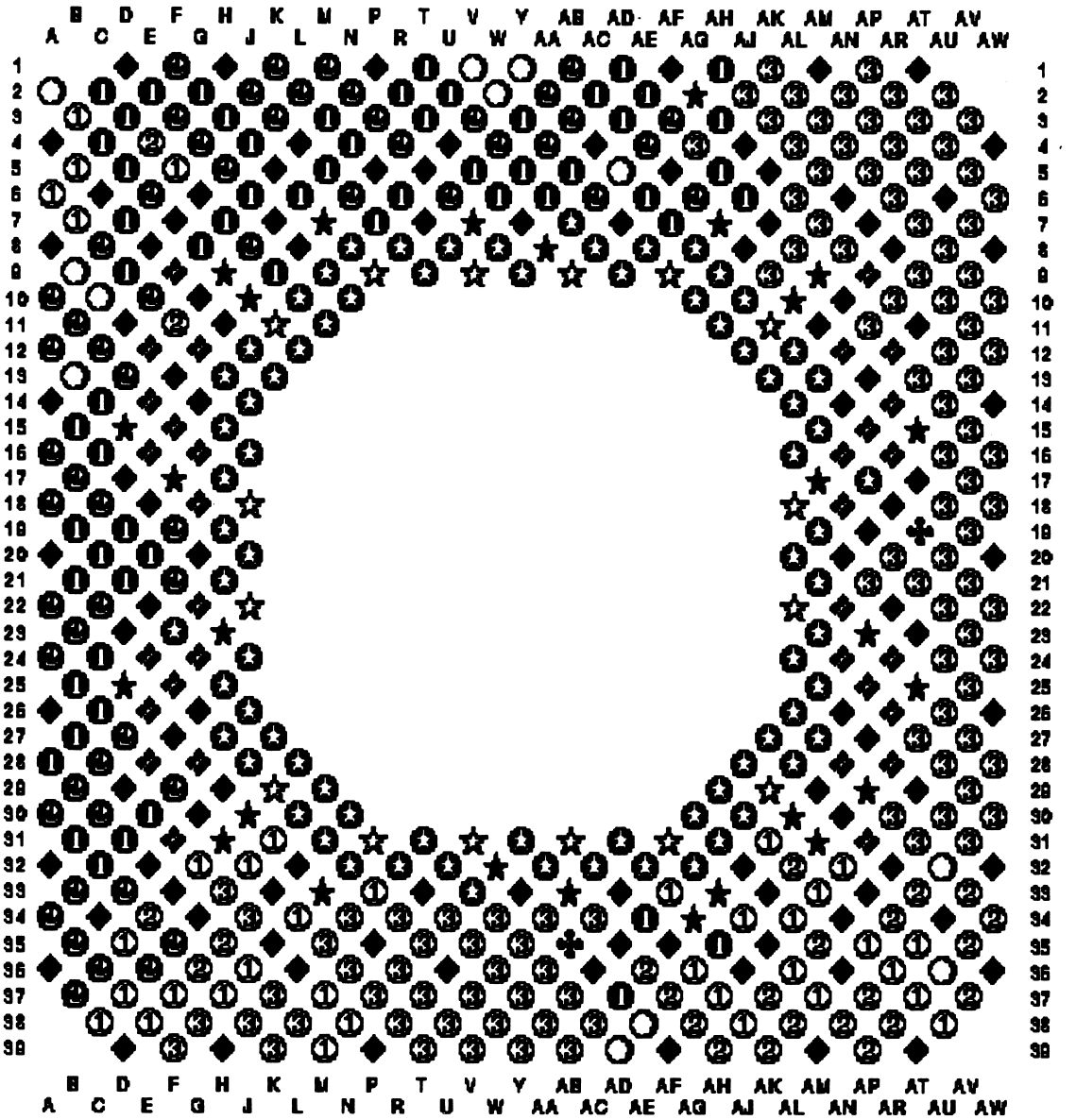
LGA557 PIN MAP FOR CCMC2 [probe-side view]



107	◆	GND	78	❶	ECL IN
92	❖	ECL GND	72	❷	ECL OUT
76	✪	VDDGA/VDDSCI	0	❸	ECL BIDIR
26	★	VTTL	8	①	TTL IN
16	☆	VECL	67	②	TTL OUT
1	○	NC	72	③	TTL BIDIR
2	✤	ANALOG			

3 MAUI

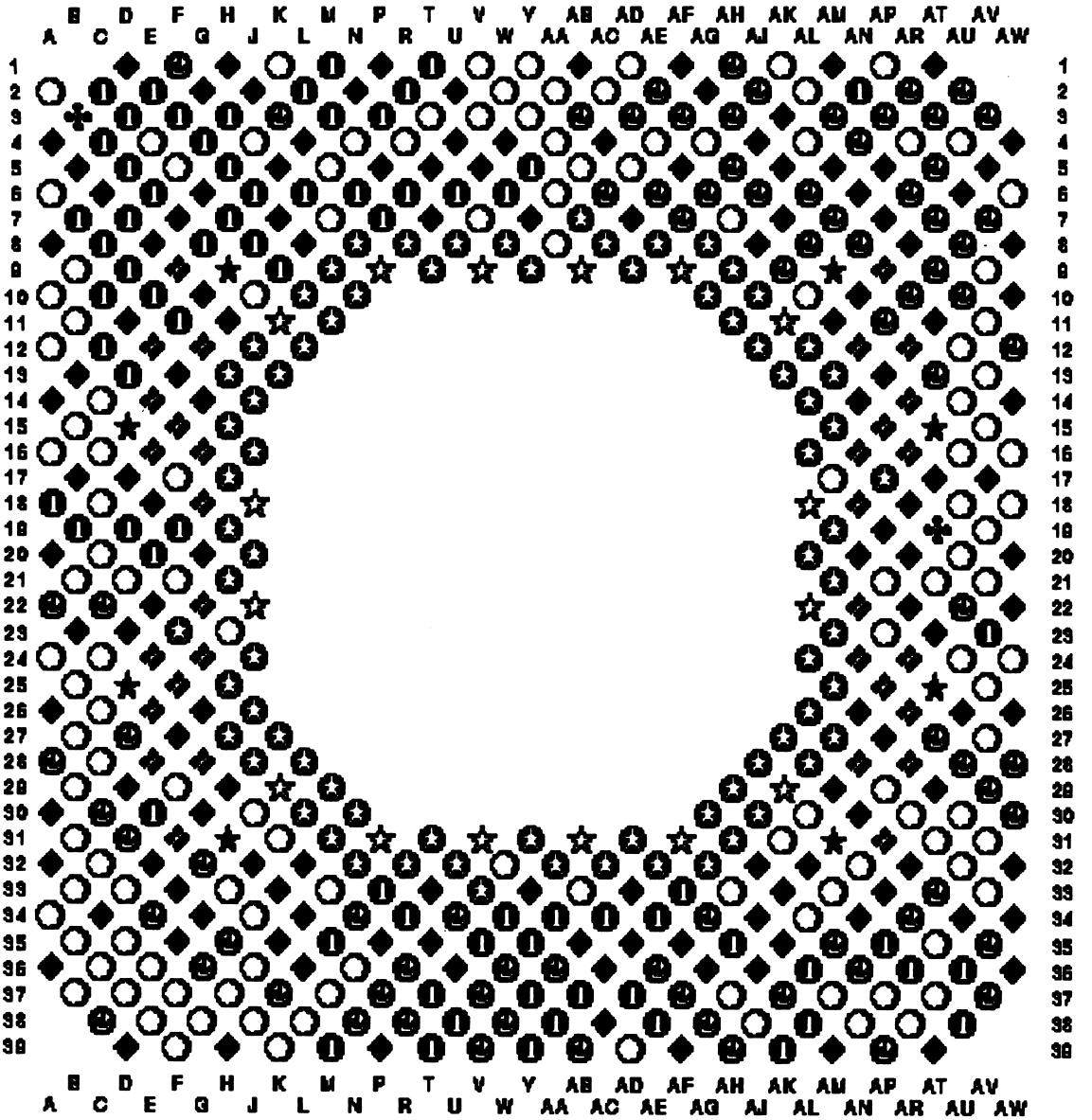
LGA557 PIN MAP FOR MAUI [probe-side view]



106	◆	GND	61	①	ECL IN
92	◈	ECL GND	57	②	ECL OUT
76	✪	VDDGA/VTTGA	0	③	ECL BIDIR
27	★	VTTL	96	①	TTL IN
16	☆	VECL	24	②	TTL OUT
12	○	NC	106	③	TTL BIDIR
2	✤	ANALOG			

4 SCI2

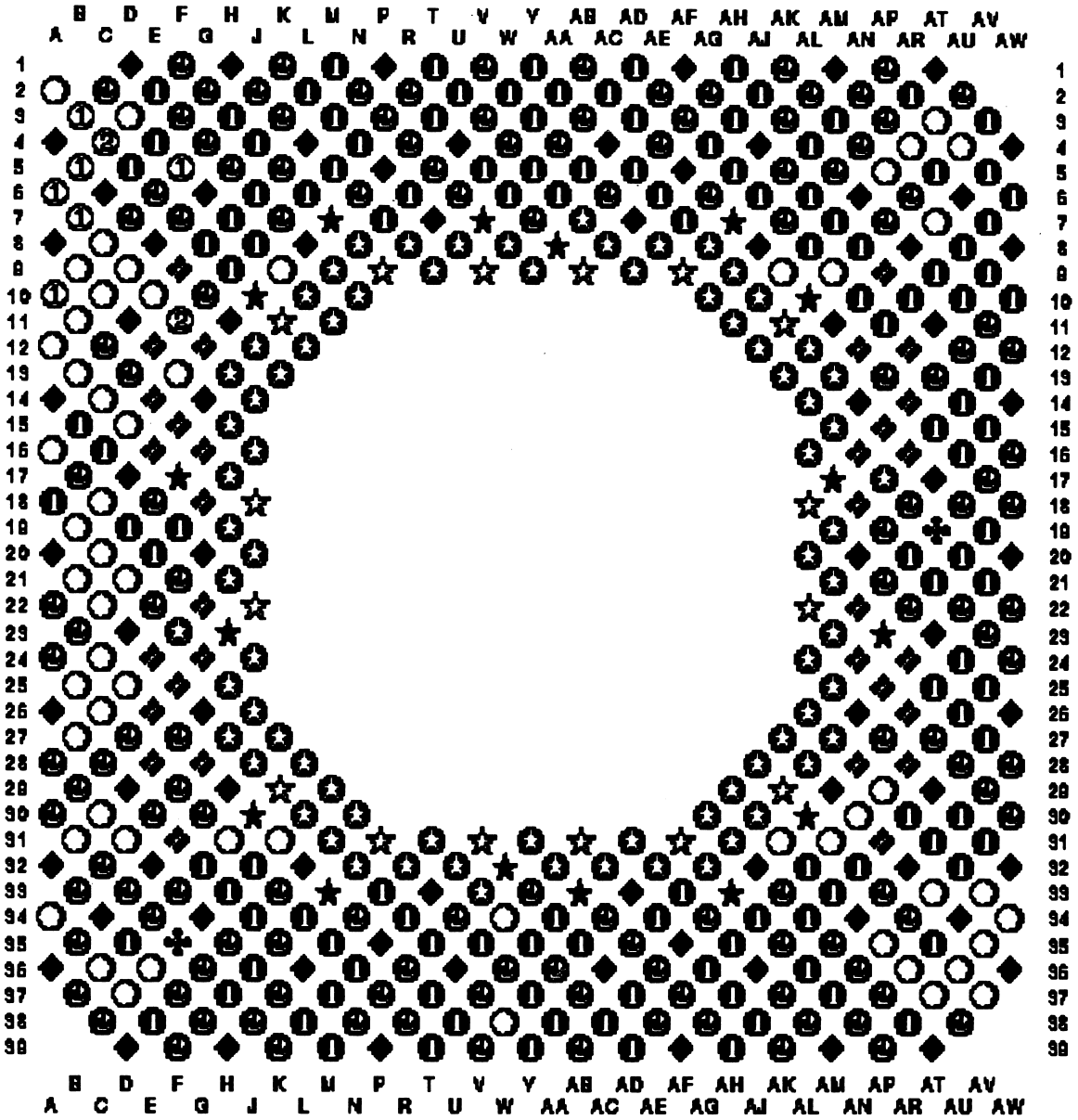
LGA557 PIN MAP FOR SCI2 [probe-side view]



197	◆	GND	73	❶	ECL IN
92	❖	ECL GND	81	❷	ECL OUT
76	✪	VDDGA/VTTGA	0	❸	ECL BIDIR
8	★	VTTL	0	⓫	TTL IN
16	☆	VECL	0	⓬	TTL OUT
192	○	NC	0	⓭	TTL BIDIR
2	✣	ANALOG			

5 XBAR

LGA557 PIN MAP FOR XBAR [probe-side view]



76	◆	GND	134	❶	ECL IN
92	❖	ECL GND	140	❷	ECL OUT
76	✪	VDDQA/VTTQA	0	❸	ECL BIDIR
16	★	VTTL	6	①	TTL IN
16	☆	VECL	2	②	TTL OUT
57	○	NC	0	③	TTL BIDIR
2	✣	ANALOG			

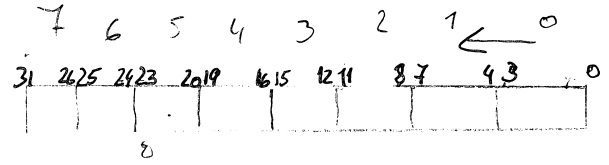
Appendix •E Quick Reference Sheet

NAME	Description	where it lives	SPP-UX safe
ccmu	Convex Configuration Utility	/spp/bin	yes*
cputest	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_cnsl	Gains control or spies the system's console	/spp/bin	yes
sppboot	Boots a node using the test station only	/spp/os	no
sppring	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	- Start_PL
29	PLSELFTEST	
28	PLCPUINIT	
27	PLINITMEM	
26	PLTESTMEM	
25	PLLOADCODE	
24	PLJUMPTOCODE	
23	PLLOADOBP	- PL_Load_obp
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	INITSCI0	
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

```
0x03ffffff
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_CLOC

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 Errors
9     HPMCH0_EN Enable HPMCH to processor 0
8     LPMCH0_EN Enable LPMCH to processor 0
7     HPMCH1_EN Enable HPMCH to processor 1
6     LPMCH1_EN Enable LPMCH to processor 1
5     HARD_HPMCH_EN Map HPMCH to Hard Errors
4     HARD_LPMCH_EN Map LPMCH to Hard Errors
3     HARD_Q_EN Q Input Port Hard Error Enable
2     HARD_S_EN S Input Port Hard Error Enable
1     HARD_INT_EN Internal Hard Error Conditions Enable
0     HARD_LOCK_FIRST Lock hard_error CSR on first enabled
error condition
```

CPA_PMON_LATENCY

```
0x00000000
```

CPA_PMON_CONTROL

```
0x00000002
```

CMC_ERR_EN0

0xfc7f3fff

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 IRD_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_OWNER_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,FlushP/PurgeP but line in use by SCI
7 CI_TRAP_ERR Attempt to access another node (enable on single node systems only)
6 REFRESH_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_RESP_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_RP 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

```
*-------------------------------------------------------------
*  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_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 -7
   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_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
```