EK-86XX1-MG-003

VAX 86XX System Maintenance Guide

For Internal Use Only

Prepared by Educational Services
of Digital Equipment Corporation

First Edition, December 1985
Second Edition, September 1986
Third Edition, March 1987

Copyright © 1985, 1986, 1987 by Digital Equipment Corporation.
All Rights Reserved.
Printed in U.S.A.

The reproduction of this material, in part or whole, is strictly prohibited. For copy information, contact the Educational Services Department, Digital Equipment Corporation, Maynard, Massachusetts 01754.

The information in this document is subject to change without notice. Digital Equipment Corporation assumes no responsibility for any errors that may appear in this document.

The following are trademarks of Digital Equipment Corporation, Maynard, Massachusetts.

[redacted]	DIBOL	Rainbow	VAX
DEC	MASSBUS	RSTS	VMS
DECmate	PDP	RSX	VT
DECUS	P/OS	RT	Work Processor
DECwriter	ProCessional	UNIBUS	

CONTENTS

PREFACE

CHAPTER 1 GENERAL INFORMATION

1.1 RELATED DOCUMENTATION 1-2
1.2 LITERATURE ORDERING INFORMATION 1-6
1.3 SOFTWARE [illegible] ORDERING INFORMATION 1-7
1.4 FIELD SERVICE MICRO-FICHE LIBRARY 1-8
1.5 RELATED INFORMATION DISTRIBUTION 1-8
1.6 VAX SUPPORT APPLICATION 1-9

CHAPTER 2 MODULE UTILIZATION AND BACKPLANE INFORMATION

2.1 VAX 8600/8650 SBI MODULE UTILIZATION . . . 2-2
2.2 VAX 8600/8650 MEMORY ARRAY MODULE CONFIGURATIONS 2-4
2.3 CABINET CAGE MODULE UTILIZATION 2-4
2.4 CPU, MBOX, AND MEMORY BACKPLANE COMPONENTS AND PIN NUMBERING SCHEMA 2-5
2.5 I/O BACKPLANE COMPONENTS AND PIN NUMBERING SCHEME 2-6
2.6 CABINET CAGE BACKPLANE - REAR VIEW 2-7
2.7 BACKPLANE INTERCONNECTIONS 2-9

CHAPTER 3 POWER

3.1 POWER SYSTEM COMPONENT DESCRIPTION 3-1
3.2 ENVIRONMENTAL SENSOR DESCRIPTIONS (AND LIMITS) 3-2
3.2.1 Air Temperature Sensors 3-2
3.2.1.1 Ambient Air Input Temperature Sensor (T1) 3-2
3.2.1.2 Module Card Cage Exhaust Temperature Sensors (T2, T3, T4) 3-3
3.2.1.3 Air Temperature Upstream Across Modules (All Sensors) 3-3
3.2.1.4 Critical Cabinet Temperature Sensor (T5) 3-3
3.2.2 Air Flow Sensors 3-3
3.2.3 Ground Current Sensor 3-5
3.2.4 50-Watt Transformer Temperature Sensor (Front End Cabinet) 3-6
3.2.5 CPU Cabinet Filter Sensor 3-6
3.3 ENVIRONMENTAL MONITORING AND EMERGENCY POWER OFF CONDITIONS 3-6
3.3.1 EMM Hardware Monitored and Controlled . . 3-6
3.3.2 EMM Software Monitored and Controlled . . 3-6
3.3.3 EMM Software Monitored and Console Software Controlled 3-8

3.3.4 BTG-A Power Controller Monitored and Controlled 3-8
3.4 REFERENCE DOCUMENTATION 3-8
3.5 SYSTEM ENVIRONMENTAL MONITORING - BUSTWARK CONTROL 3-8
3.5.1 Temperature Sensor (or Delta Temperature) Entire Yellow Zone 3-9
3.5.2 Temperature Sensor (or Delta Temperature) Entire Red Zone 3-9
3.5.3 Single Air flow Fault 3-10
3.5.4 Double Air-Flow Fault 3-10
3.5.5 BCD Status Change 3-10
3.6 MMS POWER DISTRIBUTION AND LOGIC CHARTS . . . 3-10
3.7 MMS POWER DISTRIBUTION COLOR CODE CHART . . 3-17
3.8 HOW TO MARGIN MPS REGULATORS 3-11
3.8.1 MPS Regulator Margining Procedure 3-14
3.9 H7170-A POWER SUPPLY - VISUAL INDICATORS . . 3-14
3.10 H7180-A, H7186-A, AND H7187-A POWER SUPPLY VISUAL INDICATORS 3-16
3.11 DCOAL OFF CODES - FAULT CONDITIONS 3-19
3.12 BMM LEDS AND SWITCHES 3-19
3.13 MPS POWER AND SENSOR READING 3-20
3.14 MPS STATUS MESSAGES 3-20
3.15 HOW TO ENABLE/D SABLE BATTERY BACKUP 3-20
3.16 HOW TO DETERMINE THE BATTERY CHARGE STATUS OF THE BBU 3-22
3.17 H7140-CA POWER DISTRIBUTION 3-22
3.17.1 H7140 Power Distribution Color Codes . . . 3-23
3.17.2 H7140 Power Distribution/Interconnection Tables 3-23
3.18 BL02 (DVT) POWER INFORMATION 3-24

CHAPTER 4 INITIALIZATION AND BOOTSTRAP TROUBLESHOOTING PROCEDURES

4.1 INTRODUCTION 4-1
4.2 BOOT SEQUENCE TIME LINE FLOW DESCRIPTION . . 4-1
4.3 NODULE REVIEW 4-6
4.3.1 Overview 4-6
4.3.2 Implementation 4-6
4.3.3 Parallel Key Loop 4-7
4.3.4 CPU Serial Key Loop 4-7
4.3.5 PBox Serial Key Loop 4-7
4.3.6 IOA Serial Key Loop 4-8
4.4 CCP LED SEQUENCING 4-20
4.5 CONSOLE SELF TESTS 4-22
4.6 VMS UNEXPECTED INTERRUPT TROUBLESHOOTING PROCEDURES 4-24
4.7 VMS CPU USAGE 4-24

CHAPTER 5 ERRORS AND ERROR ANALYSIS

5.1 SYSTEM FAULT ISOLATION 5-1
5.1.1 PBox Interrupt and Exception Arbitration Logic 5-1
5.1.2 Error Handling Microcode (EHM) 5-1
5.1.3 VMS Machine Check Handler 5-1
5.1.4 SPEAR (Standard Package for Error Analysis and Reporting) 5-1
5.1.5 Keep Alive Fail (KAF) 5-3

5.2 CONSOLE SNAPSHOT FILE INFORMATION 5-5
5.2.1 Guide to Decoding/Analysis of a SNAPSHOT . 5-5
5.2.2 Reference Documentation 5-4
5.2.3 SNAPSHOT File Flowchart Notes 5-4
5.3 CONSOLE STORE PARITY ERROR CORRECTION . . . 5-7
5.4 SYSTEMATIC INFORMATION GATHERING PROCEDURES 5-9
5.4.1 Method 1 5-9
5.4.2 Method 2 5-10
5.4.3 Method 3 5-10
5.4.4 Method 4 5-10
5.4.5 Method 5 5-11
5.5 DUMP.COM 5-11
5.6 PROGRAM TO SCAN MEMORY FOR PARITY ERRORS (SCAN.COM) 5-15
5.7 VMS AND VAX SYSTEM EVENT FILE (ERRLOG.SYS) . 5-15
5.7.1 ANALYZE/ERRORLOG 5-16
5.7.2 SPEAR Library Functions 5-16
5.7.2.1 SPEAR Basic 5-16
5.7.2.2 SPEAR Extended 5-17

CHAPTER 6 DIAGNOSTICS

6.1 DMF SILO TEST 6-2
6.1.1 Prerequisites 6-2
6.1.2 DMF Self Test Descriptions 6-2
6.1.3 DMF Error Code Descriptions 6-6
6.2 T-11 (PROM) SELF TEST 6-6
6.2.1 Prerequisites 6-6
6.2.2 PROM Command Set 6-6
6.2.3 T-11 (PROM) Self Test Descriptions 6-8
6.3 CONSOLE MODULE DIAGNOSTIC (KDJ8A) 6-12
6.3.1 Prerequisite 6-12
6.3.2 KDJ8A Switch Register Options 6-12
6.3.3 KDJ8A Control Characters 6-12
6.3.4 KDJ8A Test Descriptions 6-13
6.4 KDJ8A - MICRO-HARD-CORE DIAGNOSTIC (MHC) . . 6-17
6.4.1 Prerequisites 6-17
6.4.2 Loading and Starting MHC 6-17
6.4.3 Micro-Hard-Core Control Characters 6-17
6.4.4 Micro-Hard-Core Control Switches 6-17
6.4.5 Micro-Hard-Core Commands 6-18
6.4.6 Micro-Hard-Core Test Descriptions 6-20
6.5 MICRO DIAGNOSTIC CONTROL PROGRAM (DCP) . . . 6-24
6.5.1 Overview 6-24
6.5.2 DCP Control Characters 6-24
6.5.3 DCP Command Summary 6-25
6.6 VAX 8SXX MICRODIAGNOSTICS 6-30
6.7 STANDARD MICRODIAGNOSTIC COMMAND FILES . . . 6-31
6.8 DIAGNOSTIC SUPERVISOR (DSVAA) OVERVIEW . . . 6-31
6.8.1 Diagnostic Supervisor (DSVAA) Control . . . 6-34
6.9 RL02 RESIDENT VAX MACRO-DIAGNOSTICS 6-41

CHAPTER 7 SYSTEM INFORMATION

7.1 VAX 8600 TO 8650 UPGRADES 7-1
7.2 xA8600/8650 RAM AND MICROCODE LISTING INFORMATION 7-4
7.2.1 xA8600/8650 Common RAM and Microcode Listing Information 7-4
7.2.2 xA8600 Specific RAM and Microcode Listing Information 7-4

7.2.3 8N8650 Specific RAM and Microcode Listing Information 7-4
7.3 INTERRUPT LEVEL ASSIGNMENTS/SOURCES 7-6
7.4 VAX 8600/8650 SYSTEM CONTROL BLOCK 7-7
7.5 CONFIGURATION GUIDELINES 7-8
7.5.1 Kernel System 7-8
7.5.2 CPU Cabinet Expansion 7-9
7.5.2.1 CPU Backplane 7-10
7.5.2.2 Memory Backplane 7-10
7.5.2.3 Adbus Backplane 7-11
7.5.2.4 I/O Backplane 7-11
7.5.3 Front End Cabinet 7-12
7.5.4 BA11-KL/KN EXPANSION RULES 7-12
7.5.5 UNIBUS Expansion 7-14
7.5.6 SBI Expansion 7-15
7.5.6.1 SBI Configuration Rules 7-16
7.5.6.2 SBI IP Level Assignments 7-17
7.5.7 Cluster Rules 7-17
7.6 PM PROCEDURES 7-18
7.6.1 Summary of Quarterly PM Procedures 7-18
7.6.2 Summary of Annual PM Procedures 7-19
7.6.3 Quarterly PM Procedures 7-19
7.6.4 Annual PM Procedures 7-20
7.6.5 VAX8600/8650 PM Checklist 7-27

CHAPTER 8 SYSTEM CLOCKS

8.1 CLOCK INTRODUCTION 8-1
8.2 CLOCK CONSOLE COMMANDS 8-1
8.3 CLOCK OUTPUTS 8-1
8.4 CLOCK BLOCK DIAGRAMS 8-2

CHAPTER 9 CONSOLE HARDWARE

9.1 SYSTEM CONTROL PANEL 9-2
9.1.1 Console Operating Modes 9-2
9.1.2 SCP Switches and Indicators 9-7
9.2 CONSOLE (TTA) INTERRUPT AND VECTOR INFORMATION 9-8
9.3 CBUS 9-8
9.4 QBUS 9-8
9.5 SDB 9-10
9.5.1 SDB Signal Name File Information 9-14
9.5.2 CADIF File Description 9-14
9.5.3 SDB ID 9-15
9.5.4 SDB Symbol 9-17
9.5.5 SDB Signal Name 9-17
9.5.6 CADIF File Revisions and the CONFIG.DAT file 9-17
9.6 REMOTE DIAGNOSIS 9-19
9.6.1 General 9-19
9.6.2 Setting-up the Grill Modem 9-19
9.6.3 Using the Set Terminal Command 9-22

CHAPTER 10 CONSOLE SOFTWARE AND COMMANDS

10.1 CONSOLE COMMANDS 10-2
10.1.1 Control Characters 10-2
10.1.2 Console Command Syntax 10-2
10.1.3 Architecturally Defined Commands and Switches 10-3

10.2 GENERAL COMMANDS 10-3
10.2.1 The General Command Set 10-5
10.3 MACRO CONTEXT 10-15
10.3.1 MACRO Context Initialization 10-15
10.3.2 Console Support Microcode (CSM) 10-15
10.3.3 MACRO Context Command Set 10-16
10.4 IBB HSX DRIVER 10-21
10.4.1 The HSX Command Set 10-22
10.4.2 Default Visibility Registers 10-27

CHAPTER 11 EBUS

11.1 NBUS 11-1
11.1.1 NBus Signal Pin Location 11-8
11.2 EBOX GPR/SCRATCH PAD USAGE 11-9
11.3 EBOX MICROWORD 11-13
11.3.1 EBox Control Store 11-18
11.3.2 EBox Context 11-19
11.3.3 Memory Control Function (MCF) 11-19
11.4 EBOX MICROTRAP VECTOR ASSIGNMENTS 11-21
11.5 EBOX FATAL ERROR CODES 11-22
11.6 CONSOLE SUPPORT MICROCODE (CSM) 11-23
11.7 EBOX MICROCODE LOCATIONS/REGIONS 11-25
11.8 EBOX HPC TESTPOINTS 11-26

CHAPTER 12 FBOX

12.1 FBOX MICROCODE 12-6
12.1.1 FBox Adder (FBA) Microcode 12-8
12.1.2 FBox Multiplier (FBM) Microcode 12-10
12.1.3 FBox Dispatch RAM (FDRAM) 12-15
12.1.4 FBox Constitution Modules 12-15
12.1.5 Turning the FBox ON and OFF 12-16
12.1.5.1 To Disable the FBox 12-16
12.1.5.2 To Enable the FBox 12-16
12.1.5.3 To Determine the Status of the FBox . 12-16

CHAPTER 13 IBOX

13.1 IBOX MODULES 13-2
13.2 IBOX MICROCODE 13-4
13.3 IBOX DISPATCH RAM (IDRAM) 13-10
13.3.1 IDRAM Address Generation 13-11
13.3.2 IDRAM Microword 13-12
13.4 IBUFFER AND OPCODE TESTPOINTS 13-15

CHAPTER 14 MBOX

14.1 MBOX MODULES 14-7
14.2 MBOX MICROCODE 14-7
14.3 MBOX MFORK ENTRIES 14-17
14.4 MBOX CYCLE CONDITION CODE MICROWORD . . . 14-18
14.5 MBOX REGISTERS 14-22
14.6 ABUS INTERFACE 14-27
14.7 ABUS INTERFACE SIGNALS 14-31
14.8 MBOX TESTPOINTS 14-34
14.8.1 NBus MicroPC and ABus Control Testpoints 14-35
14.8.2 ABus/SBIA Testpoints 14-35
14.8.3 CP Port Testpoints 14-36

CHAPTER 15 SBIA

15.1 SBIA GENERAL INFORMATION 15-1
15.2 SBIA JUMPER SETTINGS 15-1
15.3 SBIA BLOCK DIAGRAM 15-2
15.4 VAX 8SXX PHYSICAL MEMORY ADDRESS ALLOCATION 15-3
15.4.1 SBIA Register Addresses 15-5
15.5 SBI PROTOCOL 15-6
15.6 SBI — I/O BACKPLANE INTERCONNECTIONS . . 15-10
15.7 SBI FAULT DESTRIBUTIONS 15-14
15.8 SBI CONFIRMATION AND FAULT DECISION FLOW . 15-14

CHAPTER 16 SBI

16.1 SBI NEXUS ADDRESS INFORMATION 16-1
16.1.1 SBI Nexus Addressing 16-1
16.1.2 SBI Nexus Address Conversion 16-1
16.1.3 SBI Nexus Interrupt Vector Generation . . 16-2
16.2 CI780 16-3
16.2.1 CI780 Backplane Jumper Settings 16-7
16.3 DW780 16-10
16.4 DW780 16-17
16.4.1 DW780 Jumper Settings 16-21
16.4.2 UDA 50 Module Utilization 16-23
16.4.3 Unibus Signals 16-24
16.4.4 Unibus Address to VAX Physical Address Conversion 16-28
16.4.5 VAX Physical Address (Hex) to Unibus Address (Octal) Conversion 16-28
16.4.6 DW780 Unibus Device Addresses 16-29
16.5 MH780 16-31
16.5.1 MAP Register Address Calculation 16-35
16.5.2 MA880B Register Address Calculation . . 16-35

CHAPTER 17 REVISION CONTROL

17.1 VAX 8600/8650 REVISION INFORMATION 17-1

CHAPTER 18 REMOVAL AND REPLACEMENT PROCEDURES

18.1 GENERAL 18-1
18.2 FRU PART NUMBERS 18-2
18.3 FRONT END CABINET ASSEMBLY 18-2
18.3.1 Preliminary Steps 18-3
18.3.2 Front End Cabinet Disassembly 18-5
18.3.3 Constant Voltage Transformer (CVT) Assembly Removal 18-5
18.3.4 RL02 Disk Drive Removal and Installation . 18-6
18.3.4.1 RL02 Disk Drive Removal 18-6
18.3.4.2 RL02 Disk Drive Installation 18-8
18.3.5 BA11-A Unit Assembly Removal and Installation 18-10
18.3.5.1 BA11-A Unit Assembly Removal 18-12
18.3.5.2 BA11 A Unit Assembly Replacement . . . 18-15
18.3.6 87160 Power Supply Removal 18-15
18.3.7 Fan Panel Assembly Removal (50 Hz system) 18-19
18.3.8 Front Door Filter Removal 18-20
18.3.9 Rear Door Filter Removal 18-21
18.4 CPU CABINET ASSEMBLY 18-22
18.4.1 Preliminary Steps 18-22

18.4.2 Centrifugal 1500 CFM Blower Removal . . 18-23
18.4.3 Modular Power Supply Removal 18-26
18.4.4 CPU Module Replacement 18-27
18.4.5 Array Modules 18-29
18.4.6 Air Filter Removal 18-29
18.4.7 Air Flow Sensor Removal and Installation 18-30
18.4.7.1 Air Flow Sensor Removal 18-30
18.4.7.2 Air Flow Sensor Installation 18-30
18.4.8 Air Temperature Sensor Replacement . . . 18-30
18.4.8.1 Temperature Sensor Removal 18-30
18.4.8.2 Temperature Sensor Installation . . . 18-30
18.4.9 BPC-A Power Controller Removal 18-32
18.4.10 BF231 BBU Power Supply Removal 18-34
18.5 MODULE PANEL CONNECTOR CLEANING PROCEDURE 18-36
18.5.1 Introduction 18-36
18.5.2 Required Materials 18-36
18.5.3 Precautions 18-36
18.5.4 Cleaning Procedure 18-37

CHAPTER 19 TECHNOLOGY AND TOOLS

19.1 MCA DESCRIPTORS, LOCATIONS, AND POINT SET CROSS REFERENCE 19-1
19.2 ECL TROUBLESHOOTING INFORMATION 19-6
19.2.1 Test Equipment 19-6
19.2.2 Termination 19-6
19.2.2.1 Single-ended ECL Signal Run 19-6
19.2.2.2 Bi-directional ECL Signal Run 19-6
19.2.2.3 Termination Components 19-7
19.2.2.4 How to Locate the Termination Point of a Given Signal 19-7
19.2.3 Troubleshooting ECL Signals 19-10

CHAPTER 20 VAX 8600/8650 REGISTER DESCRIPTION

EXAMPLES

6-1 End of Pass Report 6-22
7-1 Power consumption for the DMF32 7-13
9-1 Set Terminal Command 9-22
10-1 Help on Console Commands 10-4
10-2 LOAD/DUMP KA8650 10-8
10-3 SET CLOCK 10-9
10-4 SET SCSI/ECS ON 10-12
10-5 SHOW SNAP1.DAT/ASCII 10-13
10-6 X command examples 10-14
10-7 CLEAR BREAK 10-23
10-8 DEPOSIT 10-24
10-9 DEPOSIT CSPE 10-27
10-10 DEPOSIT MARK 10-27
10-11 EXAMINE CONTROL STORE 10-28
10-12 EXAMINE/CHANNEL 10-29
10-13 EXAMINE/SDS 10-30
10-14 SET BREAK 10-32
10-15 SHOW DEFINE 10-33
10-16 SHOW NAME 10-34
10-17 TRACE ADD 10-35
10-18 TRACE DEFINE 10-36
10-19 TRACE DELETE 10-36
10-20 TRACE REMOVE 10-36

11-1 Determining which CSK overlay was loaded 11-25
13-1 Generating IOBAM Addresses 13-1
16-1 Memory Address Calculation 16-2
16-2 Interrupt Vector generation 16-3
16-3 Conversion from Unibus address to VAX physical address 16-20
16-4 Calculating a physical byte address . . . 16-33
19-1 Off Module Termination 19-4
19-2 Extract from CPU Backplane Wirelist BL-Asrt (70-19198) 19-8

FIGURES

2-1 BA11-AL Module Utilization as used in VAX 8600/8650 2-1
2-2 CPU Cabinet Cage Module Utilization 2-4
2-3 CPU, ABUS, and Memory Backplane Components - Side View 2-5
2-4 I/O Backplane Pin Numbering Scheme 2-6
2-5 Cabinet Cage Backplane (Sheet 1 of 2) . . . 2-7
3-1 Power System Components 3-1
3-2 Power Shutdown Sequence 3-5
3-3 Power System Interconnect Diagram, Front End Cabinet 3-6
3-4 Power System Interconnect Diagram Power Controller 3-7
3-5 MPS Voltage Measurements 3-12
3-6 Module Power Connections 3-13
3-7 H7140-A Power Supply - Visual Indicators . 3-15
3-8 H7160-A Power Supply 3-16
3-9 H7186-A (7Y367-AA) Power Supply 3-17
3-10 H7188-A ENK Module Visual Indications . . . 3-18
3-11 ENK Module - Detailed Block Diagram 3-21
3-12 H7140-CA Power Interconnection Diagram . . . 3-23
4-1 Boot Sequence Time Line Flow 4-2
4-2 AC Input Module (K and L) Troubleshooting Flow 4-5
4-3 Clock Module Key Fault Logic 4-6
4-4 Parallel Key Loop Fault Troubleshooting Flow 4-8
4-5 Parallel Key Loop Circuit 4-9
4-6 Serial Key Loop Fault Troubleshooting Flow 4-10
4-7 CPU Key Fault Troubleshooting Flow 4-11
4-8 CPU Serial Key Loop Circuit 4-12
4-9 CBox Serial Key Loop Circuit 4-13
4-10 TIA Serial Key Loop Circuit 4-14
4-11 Regulator A (+5 Vdc) Initialization Troubleshooting Flow 4-15
4-12 Console Initialization and Self Test (1-16) Troubleshooting Flow 4-16
4-13 Console Self Test (17-35) Troubleshooting Flow 4-17
4-14 Console CBus (R1C2) Troubleshooting Flow (Sheet 1 of 2) 4-18
5-1 System Fault Termination - Functional Block Diagram 5-2
5-2 Console Snapshot File Information Flow Chart 5-5
5-3 Control Store Parity Error Simplified Flow Chart 5-8
7-1 System Block Diagram 7-2
7-2 CPU Block Diagram 7-3
7-3 Kernel System - Front View 7-9

7-4 CPU Cabinet - Front View 7-10
7-5 BA11-AA Front End Cabinet 7-12
7-6 BA11-AL/AM Backplane Diagram 7-13
7-7 UNIBUS Expansion Configurations 7-15
8-1 Clock Distribution System, L0211 Rev. C5 . 8-3
8-2 Clock Distribution System, L0231/L0217 Rev. F 8-5
8-3 L0211 Rev. C5 Clock Diagram 8-6
8-4 L0231/L0217 Rev. E Block Diagram 8-8
8-5 Backplane Clock Distribution 8-9
8-6 Clock Control Logic, CLU MCA 8-10
9-1 Console Interconnect Block Diagram 9-1
9-2 System Control Panel Switches and LEDs . . . 9-2
9-3 CBus Block Diagram 9-7
9-4 DMA Interconnect Block Diagram 9-8
9-5 SDS Interface 9-11
9-6 SDS ID Format 9-16
9-7 H112-AA Switchpack Registers 9-21
9-8 BA46 Core Distribution Panel 9-31
9-9 H112 Modem 9-32
9-10 Console Remote Port Control Flowchart . . . 9-34
10-1 Console Software Modes 10-1
10-2 Console Mode Contents 10-1
11-1 FBox Modules Block Diagram 11-2
11-2 WBus Data Path (Sheet 1 of 2) 11-4
11-3 WBus Physical Distribution 11-6
11-4 EBox Control Store Microword Worksheet . . 11-10
11-5 GBox Microword <79:80> 11-11
11-6 FBox Microword <79:64> 11-12
11-7 EBox Microword <63:48> 11-14
11-8 BBox Microword <47:32> 11-16
11-9 ABox Microword <31:16> 11-17
11-10 MBox Microword <15:00> 11-18
11-11 XBox Context (CTX) Microword Worksheet . . 11-19
11-12 MCP Microword Worksheet 11-20
12-1 KBox Basic Block Diagram 12-1
12-2 KBox Block Diagram 12-2
12-3 KBox Interface 12-3
12-4 EBox Adder Microcode Worksheet 12-4
12-5 KBA Microword <47:32> 12-4
12-6 KBA Microword <31:16> 12-6
12-7 KBA Microword <15:00> 12-9
12-8 KBox Multiplier Microcode Worksheet . . . 12-10
12-9 KBM Microword <39:22> 12-11
12-10 KBM Microword <21:16> 12-12
12-11 KBM Microword <15:00> 12-14
12-12 KBox Dispatch RAM (DRAM) Microword . . . 12-15
13-1 IBox Module Block Diagram 13-2
13-2 IBox Data Paths Block Diagram 13-3
13-3 IBox Microword Worksheet 13-4
13-4 IBox Microword <50:48> 13-4
13-5 IBox Microword <47:32> 13-5
13-6 IBox Microword <31:16> 13-7
13-7 IBox Microword <15:00> 13-9
13-8 IDRAM Microword Worksheet 13-10
13-9 IDRAM Microword <19:16> 13-12
13-10 IDRAM Microword <15:00> 13-12
13-11 Push Addresses Generation 13-13
14-1 NBox Physical Organization 14-2
14-2 NBox Address Path Block Diagram 14-3
14-3 NBox Data Path Block Diagram 14-4
14-4 NBox Interface Block Diagram 14-5
14-5 NBox Error Reporting Block Diagram 14-6
14-6 NBox Control Store Microword Worksheet . . 14-7

14-7 MBox Microword <79:64> 14-8
14-8 MBox Microword <63:48> 14-9
14-9 MBox Microword <47:32> 14-11
14-10 MBox Microword <31:16> 14-11
14-11 MBox Microword <15:00> 14-12
14-12 MBox Cycle Condition Code Microword
Highbyte 14-13
14-13 MBox Cycle Condition Code Microword <31:16> 14-13
14-14 MBox Cycle Condition Code Microword <15:00> 14-15
14-15 MBox Register Read/Write Data Paths . . . 14-22
14-16 XAD Module MBox Registers 14-23
14-17 XCI Module MBox Registers 14-23
14-18 XCD Module MBox Registers 14-24
14-19 UBox Microcode Assembly of MBox Registers 14-25
14-20 Physical Address Memory Map 14-26
14-21 SBus Interface Block Diagram 14-27
14-22 ABus CPU Command/Address Cycle Format . . 14-28
14-23 ABus DMA Command/Address Cycle Format . . 14-28
14-24 ABus Read/Write Cycle Format 14-28
15-1 SBIA Block Diagram 15-2
15-2 VAX 8600 Physical Memory Address Allocation 15-3
15-3 I/O Adapter Physical Address Allocation . . 15-4
15-4 SBI Signal Names 15-6
15-5 SBI Command/Address Format 15-8
15-6 SBI Command Codes 15-9
15-7 SBI Unibly Field Configuration 15-9
15-8 SBI Read Data Formats 15-9
15-9 SBI Write Data Format 15-10
15-10 SBI Interrupt Summary Formats 15-10
15-11 I/O Backplane Addresses 15-11
15-12 SBI Configuration and Fault Decision Flow . 15-13
16-1 SBI Unibus Address Generation 16-2
16-2 SBI Interrupt Vector Generation 16-2
16-3 CI780 I/O Backplane Module Utilization . . . 16-3
16-4 CI780 Block Diagram (Sheet 1 of 2) 16-4
16-5 CI780 Backplane Jumper Locations 16-6
16-6 DR780 Module Utilization 16-10
16-7 DR780 Block Diagram (Sheet 1 of 4) 16-11
16-8 DR780 Backplane Jumper Location 16-15
16-9 DW780 I/O Backplane Module Utilization . . 16-17
16-10 DW740 Expander Cabinet Module Utilization 16-18
16-11 DW780 Block Diagram 16-19
16-12 DW780 I/O Backplane Jumper Locations . . . 16-20
16-13 UDA 50 Module Utilization 16-23
16-14 Unibus Signals 16-24
16-15 DH780 Module Utilization 16-31
16-16 DH780 Block Diagram (Sheet 1 of 2) 16-32
16-17 DH780 Jumper Settings 16-34
16-18 Map Register Address Calculation 16-35
16-19 VAXBUSR Register Address Calculation . . . 16-36
17-1 System Control Panel 18-3
18-2 BTE-A Power Controller - Front View 18-5
18-3 Front Mod Cabinet Cable Connections - Rear
View 18-4
18-4 RL01 Disk Drive - Rear View 18-6
18-5 RL02 Disk Drive Mounted in Slides 18-7
18-6 Slide Mounting Rail and Slide 18-7
18-7 RL02 Disk Drive with Exposed Drive Logic
Module 18-9
18-8 RL02 with Covers Removed 18-10
18-9 RL02 Disk Drive - Front View 18-11
18-10 Cable Clamps 18-12
18-11 Release Latch 18-13

18-12 BA[1-5 Installed in Slide Mount 18-13
18-13 Slide to Index Plate Mounting 18-14
18-14 Power Supply Unit, Rear Mounting Screws . 18-16
18-15 Power Supply Unit Removal 18-17
18-16 Mounting Box, Power Cable Connection . . . 18-18
18-17 Fan Panel Assembly 18-19
18-18 Front End Cabinet - Front Door Filters . . 18-20
18-19 Front End Cabinet - Heat Exchanger Filters . . 18-21
18-20 System Control Panel 18-22
18-21 CPU Cabinet - Top Cover Removal 18-23
18-22 CPU Blower Cable Removal 18-24
18-23 CPU Blower Removal - Rear View 18-25
18-24 Modular Power Supply Removal 18-26
18-25 Module Case Ground Connection 18-27
18-26 CPU Module Access Gate 18-28
18-27 CPU Card Cage Air Filter Removal 18-29
18-28 Temperature and Air Sensor Locations . . . 18-31
18-29 Power Controller and BBU - Rear View . . . 18-32
18-30 Power Controller Removal 18-33
18-31 BBU Cabling 18-34
18-32 BBU Removal - Top View 18-35
18-33 Paddle Finger Cleaning Action 18-37
19-1 Single-ended XCL Signal Run 19-6
19-2 Bi-directional XCL Signal Run 19-7
19-3 XTERM Termination 19-8
19-4 YTERM Termination 19-8
19-5 ONCE A B C Signal Generation 19-9
19-6 ONCE A & C Signal Termination. 19-9
19-7 Typical XCL Waveform 19-10
19-8 Flowchart, XCL Signal Troubleshooting . . 19-11
19-9 Improper Termination: Too Low 19-13
19-10 Output Shorted to +2V 19-12
19-11 Unterminated Waveform, 56 Ohm Resistor Missing 19-13
19-12 Inverted Output Shorted to - 2V 19-13
19-13 Improper Termination: Too High 19-14
19-14 Capacitor in Signal Run 19-14
19-15 Open Etch, No Connection to - 2V 19-15
19-16 Impedance Mismatch 19-15
19-17 Signal Shorted to Ground 19-16
19-18 Open Input Etch 19-16
19-19 Two Shorted Signals 19-17
19-20 Complementary Outputs Shorted Together . . 19-17

TABLES

6-1 EXM Self Test 6-2
6-2 EXM Error Code Descriptions 6-6
6-3 PMOX Command Set 6-7
6-4 T-11 (PXXM) Self Test Descriptions 6-8
6-5 Micro-Hard-Core Commands 6-15
6-6 Clock Subtests 6-20
6-7 RL02 Disk Subtests 6-21
6-8 XBox Subtests 6-21
6-9 IBox Subtests 6-21
6-10 Test Box Subtests 6-22
6-11 MBox Logic Subtests 6-22
6-12 MBox Ucode Subtests 6-22
6-13 MCP, CTX, and MIEC BBox Subtests 6-22
6-14 XBox, SDD, and C/S Subtests 6-23
6-15 BBox Ucode Subtests 6-27

6-16 VAX 6xxx Microdiagnostics 6-32
6-17 Standard Microdiagnostic Command Files . . . 6-33
6-18 MMU/Memory Quick Verify Command Files . . . 6-33
6-19 CPU Module Quick Verify Command Files . . . 6-34
6-20 Diagnostic Supervisor Control Character Summary 6-35
6-21 Diagnostic Supervisor Command Summary . . . 6-35
6-22 Device List for ATTACH, SELECT, and DESELECT Commands 6-41
6-23 XLO2 Resident VAX Macrodiagnostics 6-42
8-1 MBus Enable Signals 8-2
8-2 Clock Reset Signals 8-2
9-1 System Control Panel Indicators 9-3
9-2 System Control Panel Switch Summary 9-4
9-3 TLI Interrupt Vectors 9-5
9-4 CBus Signal Description 9-6
9-5 CBus Signal Backplane Pin Location 9-7
9-6 QBus Signal Backplane Pin Locations 9-8
9-7 QBus Signal Descriptions 9-9
9-8 SDB Interconnect Chart A - SCP Interface . . 9-11
9-9 SDB Interconnect Chart B - SDB Control Signals 9-12
9-10 SDB Interconnect Chart C - SDB Clock and Data Out Connections 9-13
9-11 SDB IC Bit Descriptions 9-16
9-12 VAX 8600 CADIT File Revision Information . . 9-18
9-13 VAX 8650 CADIT File Revision Information . 9-19
9-14 Set Terminal Syntax and Switch Description . 9-22
10-1 Console Command Control Characters 10-2
10-2 Architecturally Defined Commands and Switches 10-3
10-3 General Commands 10-3
10-4 Default System Microcode 10-5
10-5 Console Flags 10-7
10-6 MACRO Context Commands 10-16
10-7 Supported CPR Names 10-17
10-8 Supported IPR Names 10-18
10-9 Supported IR Names 10-19
10-10 Supported Miscellaneous Register Names . . 10-19
10-11 The SHX Command Set 10-22
10-12 SDB Control Channels 10-24
10-13 Control Channel Bit Positions 10-24
10-14 SDB Channel Numbers 10-28
10-15 Names of Default Visibility Registers . . 10-30
10-16 Default Visibility Registers 10-37
11-1 MBus Signal Pin Location 11-6
11-2 Distribution of MBus Byte Parity Signals . . 11-8
11-3 EBox GPR/Scratch Pad Usage 11-9
11-4 Context (Ctx0) 11-19
11-5 KCP Microword Bit Description 11-20
11-6 EBox Microtrap Vectors 11-21
11-7 EBR Fatal Error Loops 11-22
11-8 Special CSM Microaddresses 11-23
11-9 CSM Overlays 11-24
11-10 EBox Microcode Locations/Regions 11-25
11-11 EBox Micro PC Testpoints 11-26
13-1 TURAM Address Generation 13-11
13-2 IBUFFER Testpoints 13-15
13-3 UP UXDR Testpoints 13-16
14-1 MBox Branch Conditions 14-16
14-2 MBox Entries in MPDRR 14-17
14-3 ABus Signal Pin Location 14-29
14-4 ABUS Commands 14-31

14-5 Length/Status for CPU Command/Address Cycle 14-31
14-6 Length/Status for CMA Command/Address Cycle 14-32
14-7 Address Control of the DC032 Register File 14-33
14-8 DC032 Register File Address Control . . . 14-34
14-9 MBox MicroPC and Adam Control Testpoints . 14-35
14-10 ABUS/SBIA Testpoints 14-35
14-11 CP Port Testpoints 14-36
15-1 SBIA DC101 TR Jumpers 15-1
15-2 SBIA Register Addresses 15-5
15-3 SBI Signal Names and Description 15-7
15-4 SBI Cable Interconnections 15-10
15-5 SBI Signal and Portplane Pins 15-12
15-6 SBI Fault Definitions 15-14
16-1 CBI Nexus Base Address 16-1
16-2 CBI Nexus Address Generation Bit Descriptions 16-2
16-3 SBI Vector Generation Bit Description . . . 16-2
16-4 C1780 Backplane Jumpers 16-7
16-5 DR780 Backplane Jumper Settings and Mirroring Selection 16-16
16-6 DW780 Jumper Settings 16-21
16-7 Unibus Signal Descriptions 16-24
16-8 Unibus Pin Assignments - Standard and Modified 16-27
16-9 DW780 Unibus Device Addresses 16-29
16-10 MBA Base Addresses 16-36
16-11 MBA/Unib Adapter Offset 16-36
17-1 VAX 8600 Revision Information 17-2
17-2 VAX 8650 Revision Information 17-4
17-3 VAX 8600 Diagnostic Media Revision Information 17-6
17-4 VAX 8650 Diagnostic Media Revision Information 17-6
19-1 MCA/Print Set Cross Reference 19-1
19-2 MCA Descriptions and Locations 19-4

PREFACE

The purpose of this guide is to provide a source of operating, maintenance, programming, and troubleshooting information for the VAX 8600/8650 System that is frequently referenced by DIGITAL Field Service, Manufacturing, Training, and Engineering personnel.

The main differences between this release and the previous release are as follows:

1. The format has been changed back to one volume.
2. The manual has been updated to include the M8706-DA.
3. The CSR.SYNDS register has been added to Chapter 20, Register Description.
4. EBSR <7>, MBOX TRAP TO 4, was added to format the SMSR register.
5. The syndrome was added to the description of MCESC <14:09>.
6. MSTAT2 <26,24> was updated to include the M8706-DA.
7. Minor technical errors have been corrected.

The following paragraphs provide a brief description of the contents of each chapter.

Chapter 1 General Information

This chapter contains a list of documents, documentation ordering information, microfiche information, and related publication information.

Chapter 2 Module Utilization and Backplane Information

Chapter 2 contains module locations within the CPU cabinet, front and cabinet and other equipment cabinets.

Chapter 3 Power

This chapter contains information on module keying, the power system, and the SMM.

Chapter 4 Power-up Initialization and Bootstrap Troubleshooting Procedures

This chapter provides a description of the events that occur during power-up and the bootstrap sequence. It also includes a set of troubleshooting procedures to aid in isolating faults that may occur during power up and the bootstrap sequence.

Chapter 5 Errors and Crash Analysis

This chapter includes an overview of system fault isolation, console snapshot file information, and a simplified flowchart of ROM's control store parity error handling. Also included is systematic information gathering procedures, a list of CSH overlays, DCMP.CSM, and a program to scan for memory parity errors.

Chapter 6 Diagnostics

This chapter contains a description of the RMM and 11-11 self tests, Micro Hard Core tests, Micro Diagnostics, Macro Diagnostics, PXCM commands and the commands for each of the diagnostic contexts.

Chapter 7 System

This chapter contains basic system block diagrams and system configuration guidelines.

Chapter 8 Clock

This chapter contains detailed clock block diagrams and clock signal information.

Chapter 9 Console Hardware

Included in this chapter are console block diagrams, system control panel descriptions, CBus and QBus information, and information pertaining to remote diagnosis.

Chapter 10 Console Software and Commands

This chapter contains the console commands for the various console contexts other than the diagnostic contexts.

Chapter 11 ZBox

This chapter includes ZBox module utilization, block diagrams, signal distribution, and microcode information.

Chapter 12 YBox

This chapter includes YBox module utilization, block diagrams, and microcode information.

Chapter 13 IBox

This chapter includes IBox module utilization, block diagrams, and microcode information.

Chapter 14 MBox

Chapter 14 contains information pertaining to the MBox module utilization, block diagrams, and microcode, as well as AMux signals and protocol.

Chapter 15 SBIA and SBI

This chapter provides the SBIA and SBI module utilization, SBIA block diagrams and jumpers, and SBI signals and protocol.

Chapter 16 SBI Nexus and Internal Options

Chapter 16 contains information on the CI780, DW780, DW750, and Unibus, including jumper installation. NAD register and Nexus register address calculations are explained.

Chapter 17 Revision Control

Chapter 17 is a reduction and lift from the revision matrix to provide various CPU backplane and module revisions.

Chapter 18 Removal and Replacement Procedures

This chapter provides FRU part numbers and removal and replacement procedures for the front and cabinet and CPU.

Chapter 19 Technology and Tools

Chapter 19 contains MCL and RAM cross reference, part number, and print set information as well as normal and nonnormal ECL waveforms.

Chapter 20 Registers

This chapter contains the register bit maps and register bit descriptions.

CHAPTER 1

GENERAL INFORMATION

1.1 RELATED DOCUMENTATION

DOCUMENT TITLE	ORDER NUMBER
BA11-A Mounting Box and Power Technical Manual	EK-BA11A-TM
BA11-A Unit Assembly Illustrated Parts Breakdown (IPB)	EK-BA11A-IP
C1780 Documentation Kit	QB-C1780-GD
C1780 F.S. Maintenance Print Set	MP-01267
C1780 Technical Description	EK-C1780-TD
C1780 Users Guide	EK-C178C-UG
Communications Option Mini Reference Manual	EK-CMINI-RM
DSV11-A Bus Repeater Change Notice	EK-DSV1A-T1
DSV11-A Bus Repeater Manual	EK-DSV1A-TM
DELNI ETHERNET Local Network Interconnect Manual	EK-DELNI-TM
DELNI Installation Owners Manual	EK-DELNI-IM
DELNI Local Network Interconnect (LNI)	EK-DNLNI-IP
DELNI Rackmount Option Installation Guide	EK-DELNX-IN
DESRP-AA Local ETHERNET Repeater Manual	EK-DESRP-TM
DESRP-AA Remote ETHERNET Repeater Manual	EK-DESRP-IM
DEUNA F.S. Maintenance Print Set	MP-01378-00
DEUNA Technical Manual	EK-DEUNA-TM
DEUNA Technical Manual Update	EK-DEUNA-T2
DEUNA Users Guide	EK-DEUNA-UG
DF03 Modem Family Technical Manual	EK-DF03-TM
DF03 Power Unit Assembly (PUA)	EK-DF03-IP
DF03 Modem Users Guide	EK-DF03-UG
DF112 User Guide	EK-DF112-UG
DH11 Preventive Maintenance Manual	EK-DH11-PM
DH11 Technical Manual	EK-DH11-TM
DH11 Users Guide	EK-DH11-UG
DHU11 Interface User Guide	EK-DHU11-UG
DMF32 F.S. Maintenance Print Set	MP-00965-00
DMF32 Maintenance Advisory	AK-DMF32-MA
DMF32 Multi Function Change Notice	EK-DMF32-T1
DMF32 Users Guide	EK-DMF32-UG

DOCUMENT TITLE	ORDER NUMBER
DMF Synchronous Controller Technical Manual	EK-DMF32-TM
DMF11 Synchronous Controller User Guide	EK-DMF11-UG
DMR-11 Users Guide	EK-DMR11-UG
DMB11 Synchronous Controller Technical Manual	EK-DMB11-TM
DM832 F.S. Maintenance Print Set	MP-00699-01
DM832 User Guide	EK-DM832-UG
DM832 User Guide Update Notice	EK-DM832-UU
DR11-C General Device Interface Manual	EK-DR11C-TM
DR11-C General Device Users Manual	EK-DR11C-UG
DR11-C Preventive Maintenance Manual	EK-DR11C-PM
DR11-W Direct Memory Users Manual	EK-DR11W-UG
DS700 Documentation Kit	MP-00700-00
DS760 F.S. Maintenance Print Set	MP-00695-00
DS780 General Purpose Technical Manual	EK-DS780-TD
DS780 General Purpose Users Guide	EK-DS780-UG
DUP11 BSC Synchronous User Change Guide	EK-DUP11-01
DUP11 BSC Synchronous Change Notice	EK-DUP11-KJ
DUP11 Interface Users Manual	EK-DUP11-UP
DUP11 Maintenance Manual	EK-DUP11-MM
DV780 Documentation Kit	EK-DV780-DK
DV780 Field Service Maintenance Prin. Set (ESNPS)	MP-00497-00
DV700 Technical Description Manual	EK-DV780-TD
DZ11 Asynch MX Technical Manual	EK-DZ11D-TM
DZ11 Users Guide	EK-DZ11D-UG
DZ32 Asynchronous Multiplexer Change Sheet	EK-DZ32-CI
DZ32 Asynchronous User Change Notice	EK-DZ32-UU
DZ32 Technical Manual	EK-DZ32-TM
ETHERNET Communications Server Installation Guide	EK-DECSA-IN
ETHERNET Installation Guide	EK-ETHER-IN
ETHERNET Operation and Maintenance	EK-DECSA-OP
ETHERNET Port Tester Technical Manual	EK-DPT3S-TM
ETHERNET Repeater Technical Manual	EK-DER87-TM
H4000 ETHERNET Transceiver Installation Manual	EK-H4000-IN
H4000 DEC ETHERNET TRANSCEIVER Technical Manual	EK-H4000-TM
H9642 Series Cabinets Users Guide	EK-H9642-UG
HSC50 Field Service Maintenance Print Set (FSNPS)	MP-01422-00
HSC50 Installation Manual	EK-HSC50-IN
HSC50 Maintenance Guide	AA-FA726-TK
HSC50 Service Manual	EK-HSC50-SV
Introduction to VAX/VMS for Field Service Workbook	EY-00160-00
VAX 8600/8650 System Description and Processor Overview	EK-KA865-TD
VAX 8600/8650 System Power Technical Description	EK-KA86P-TD
VAX 8600/8650 Console Technical Description	EK-KA860-TD
VAX 8600/8650 EMA Technical Description	EK-KA860-TD
VAX 8600/8650 System Clocks Technical Description	EK-KA86K-TD
VAX 8600/8650 SBox/Memory Technical Description	EK-KA86X-TD

DOCUMENT TITLE	ORDER NUMBER
VAX 8600/8650 IBox Technical Description	EK-KA86I-TD
VAX 8600/8650 KBox Technical Description	EK-KA86K-TD
VAX 8600/8650 FBox Technical Description	EK-FPU8X-TD
VAX 8600/8650 SBIA Technical Description	EK-SBI8X-TD
VAX 8600/8650 System Diagnostics User's Guide	EK-KA8[illegible]-UG*
VAX 8600 Console Commands Reference Card	EK-KA86D-RC*
VAX 8600/8650 System Hardware User's Guide	EK-8600H-UG
VAX 8600/8650 System Maintenance Guide	EK-86XV1-MG*
VAX 8600/8650 System Installation Manual	EK-8600I-IN
VAX 8600/8650 System Fault Isolation Manual	EK-8600E-FM*
VAX 8600/8650 Field Service Maintenance Print Set	MP-01714-00*
VAX 8600/8650 Customer Print Set	MP-01714-00
VAX 8600 Field Service 8650 Upgrade Print Set	MP-01990-01*
VAX 8650 Customer 8650 Upgrade Print Set	MP-01990-00
VAX 8600/8650 Illustrated Parts Breakdown (IPB)	EK-VENU8-IP
VAX 8600/8650 Advanced Maintenance Course	EY-1402E-001
VAX 8600/8650 Instructor Guide Package	EY-1402K-IA
VAX 8600/8650 Instructor Guide	EY-1402E-IG
VAX 8600/8650 Transparency Masters	EY-1402K-0T
VAX 8600/8650 Answer Sheet/Solutions	EY-1402E-TA
VAX 8600/8650 Student Guide Package	EY-1402E-IE
VAX 8600/8650 Student Guide	EY-1402K-SG
KMS-11/DP Synchronous Comm Processor User Guide	EK-KMS8E-UG
KMS11 Synchronous Communication Proc Service Guide	EK-KMS11-PS
KMS11-P Synch Comm Processor Installation Manual	EK-KMSIP-IN
KMS11-P Synchronous Comm Proc Technical Manual	EK-KMSIP-TM
LA100 Head Lift Kit Installation Manual	EK-L1CEP-IN
LA100 Programmer Reference Manual	EK-LA100-RM
LA100 Series Pocket Service Guide	EK-LA100-PS
LA100 Letter/Writer Serial Printer (IPB)	EK-LA100-IP
LA100 Series Technical Manual	EK-LA100-TM
LA12 DECwriter Correspondent (IPB)	EK-CPL12-IP
LA12-CA/LA12X-CB Installation Guide	EK-L12XA-IN
LN01 Electronic Printer Functional Description	EK-LN01S-TD
LN01 Electronic Printer Inst Guide	EK-LN01S-IN
LN01 Electronic Printer Oper Flip Card	EK-LN01S-RC
LN01 Electronic Printer Operator Guide	EK-LN01S-OP
LN01 Programmer Reference Manual	EK-LN01S-RM
LN01S Electronic Printer Documentation Pkg	EK-LN100-DC
LP11/LS11 Line Printer Manual	EK-OLP11-TM
LP11/LS11/LA11 Line Printer Manual	EK-ILP11-TM
LP11/K8/A11 Line Printer Users Guide	EK-LP11S-OP
LP27 Line Printer Installation Manual	EK-OLP27-IN
LP27 Line Printer Technical Manual	EK-OLP27-TM
LP27 Line Printer Users Guide	EK-OLP27-OG
LP27 Pocket Service Guide	EK-OLP27-PS
LP27 Users Guide Update	EK-OLP27-01
LPA11-K Lab Installation and Maintenance Guide	EK-LPA11-IN
LPA11-K Users Guide	EK-LPA11-UG

DOCUMENT TITLE	ORDER NUMBER
LXY12/LXY22 Impact Printer/Plotter Install Guide	EK-LXY22-IN
LXY12/LXY22 Impact Printer/Plotter User Guide	EK-LXY22-OG
PCL11-B System Options Pocket Guide	EK-PCL1B-PS
PCL11-B Installation Manual	EK-PCL1B-IN
PCL11-B Technical Manual	EK-PCL1B-TM
PDP-11 Bus Handbook	EB-17525-20
RA60 Disk Drive Illustrated Parts Breakdown (IPB)	EK-ORA60-IP
RA60 Disk Drive Service Manual	EK-ORA60-SV
RA60 Disk Drive Technical Change Notice	EK-ORA60-TC
RA60 Disk Drive Users Guide	EK-ORA60-UG
RA60 Field Service Maintenance Print Set (FSMPS)	MP-01421-00
RA60 Maintenance Guide	AZ-N680A-IC
RA60 Safety Manual	EK-ORA60-HS
RA60 Subsystem Fault Isolation Guide	EK-ORA60-FG
RA60 Tech Doc Change Notice	EK-ORA60-SU
RA80 Disk Drive Change Notice	EK-ORA80-UU
RA80 Disk Drive Service Manual	EK-ORA80-SV
RA80 Disk Drive User Guide	EK-ORA80-UG
RA80/BOA80 Disk Drive Illustrated Parts Breakdown (IPB)	EK-ORA80-IP
RA81 Customer Equip Care	EK-ORA81-EC*
RA81 Disk Drive Illustrated Parts Breakdown (IPB)	EK-ORA81-IP
RA81 Disk Drive Service Manual	EK-ORA81-SV
RA81 Disk Drive User Guide	EK-ORA81-UG
RA81 Field Service Maintenance Print Set (FSMPS)	MP-01359-00
RA81 Documentation Guide	AA-N679A-TC
RE780 Documentation Kit	MK-RE780-00
RE780 Technical Description Manual	EK-RE780-TD
RL01/02 P.M. Worksheet	EK-RLU12-WS*
RL01/02 Pocket Service Guide	EK-RLU12-PS
RL01/02 Users Guide	EK-RL012-UG
RL01/02 Technical Manual Vol 1	EK-RL121-TM
RL01/02 Technical Manual Vol 2	EK-RL122-TM
RL02 Disk Drive Illustrated Parts Breakdown (IPB)	EK-ORL02-IP
RL02 Documentation Kit	MB-RL02-00
RL11 Controller Technical Description	EK-ORL11-TD
RM05 Disk Drive Illustrated Parts Breakdown (IPB)	EK-ORM05-IP
RM05 Disk Drive Maint Manual	EK-ORM05-MG*
RM05 Disk Drive Service Manual	EK-ORM05-SV
RM05 Disk Subsystem Users Guide	EK-ORM05-UG
RM05 Fault Isolation Guide	EK-ORM05-FG
RM05 P.M. Worksheet	EK-ORM05-WS*
RP07 Documentation Kit	MB-RP07-00
RP07 Disk Drive Customer Equip Care	EK-ORP07-EC*
RP07 Disk Drive P.M. Worksheet	EK-ORP07-WS*
RP07 Field Maintenance Print Set #2	EK-ORP07-MP
RP07 Illustrated Parts Breakdown	EK-ORP07-IP
RP07 Microcode Listing	EK-ORP07-ML
RP07 Pocket Service Guide	EK-ORP07-PS
RP07 Service Manual	EK-ORP07-SV
RP07 Technical Description Manual	EK-ORP07-TD
RP07 Users Guide And Addendum	EK-ORP07-UG
RP07 Error Codes Troubleshooting Manual	EK-ORP07-HP

DOCUMENT TITLE	ORDER NUMBER
ACUCS User Supplied Users Guide	EK-ACUCS-UG
TA78 Magnetic Tape Drive Manual	EK-OTA78-MV
TA78 Magnetic Tape Drive User Guide	EK-OTA78-UG
TA78 Upgrade Procedure	EK-TA78U-IN
TE16 DEC Magtape Illustrated Parts Breakdown (IPB)	EK-OTE16-IP
TE16 Vocker Service Guide	EK-OTE16-SG
TE16/TE16M/TE16N Equipment Care	EK-OTE16-EC*
TE16/TE16M/TE16N Maintenance Manual	EK-OTE16-MN
TE16/TK10N/TE10N User Change Notes	EK-OTEMK-UC
TE16/TE10M/TK10M Users Manual	EK-OTEMK-OP
TU52 Formatter Illustrated Parts Breakdown (IPB)	EK-OTU52-IP
TU78 Magnetic Tape Formatter Tech	EK-OTU78-TM
TU77 Documentation Kit	EK-TU77-DC
TU77/78 R.M. Workshop	EK-TU77U-WS*
TU77/78 Magnetic Tape Transport Equip Spec	EK-TU77S-SP
TU77 Magnetic Tape Transport Tech Manual V1	EK-TU77-TM
TU77 Magnetic Tape Transport (IPB)	EK-OTU77-IP
TU77 Magnetic Tape Transport Tech Manual V2	EK-TU77-TM
TU77 Magnetic Tape Transport Users Guide	EK-OTU77-OG
TU77 Subsystem Pocket Service Guide	EK-OTU77-PS
TU78 Documentation Kit	EK-TU78-DC
TU78 Magnetic Tape Transport Addendum	EK-TU78-12
TU78 Magnetic Tape Transport Tech Manual V1	EK-TU78-TM
TU78 Magnetic Tape Transport Users Guide	EK-OTU78-UG
TU78 Magnetic Tape Transport Tech Manual V2	EK-TU78-TM
TU78 Subsystem Pocket Service Guide	EK-OTU78-PS
TU78/TM78 Mag Tape Transport (IPB)	EK-OTU78-IP
TU80 Tape Drive Illustrated Parts Breakdown (IPB)	EK-OTU80-IP
TU80 Market Service Guide	EK-OTU80-MS
TU80 Subsystem Users Guide	EK-OTU80-UG
TU80 Tape Subsys Technical Manual	EK-OTU80-TM
TU81 Mag. Tape FS Guide	EK-OTU81-FS
TU81/TA81 Pathfinder	EK-TUA81-PF
TU81/TA81 Tape Subsys User Guide	EK-TUA81-UG
UDA50 Field Service Maintenance Print Set (VSMAIL)	EK-OUDA0-FM
UDA50 Programmer's Documentation Kit	QA905-GZ
UDA50 Maintenance Guide	EK-MUA50-TG
UDA50 Service Manual	EK-OUDA0-SV
UDA50 User Guide	EK-OQUD0-UG
VAX Architecture Handbook	EB-19563-20
VAX Hardware Handbook	EB-21710-20
VAX Software Handbook	EB-21713-20
VAX Maintenance Handbook, VAX Systems	EK-VAXVL-HB
VAX Maintenance Handbook, VAX-11/750	
VAX Maintenance Handbook, VAX-11/780	EB-VAXV2-HB
VAXcall Processing Card	AV-DS27C-TB
VAX-11 Architecture Reference Manual	EK-VAXAR-RM

DOCUMENT TITLE	ORDER NUMBER
VAX/VMS Document Set	QA001-GZ
VAX/VMS Quick	QLXY1-GZ
VAX/VMS Quick Reference Guides	QLXY2-GZ
VAX/VMS Reference Shelf	QLZ20-GZ
VAX/VMS Internals and Data Structures	Digital Press
VAX/VMS License & Warranty (Includes VAX 9000/6000 Software Installation Guide)	QA001-GZ

NOTES

1. IPB means Illustrated Parts Breakdown

2. An asterisk (*) denotes a Customer Field Service use only.

1.2 LITERATURE ORDERING INFORMATION

PREFERRED METHOD:

Use Form: EN-01218-05, Request for Literature/Technical Documents

Mail to:

Interdepartmental Mail: MRO2/WU3, Mail Order Processing

US Mail: Digital Equipment Corporation
10 Forbes Road
Northboro, MA 0 532
Attn: Mail Order Processing

ALTERNATE METHODS

Required Information: All requests submitted on other than the standard form must contain the following information:

Name:
Order Number
Cost Center: (3-letter code)
Location: (mail stop)
Ship To Information (if different from requisitioners location)

Please Use the Following Format:

Item	Quantity	Publication #	Title/Description
1.	10	EK-XE1CL-MP	Maintenance Manual
2.	100	EM0-12450-05	Medical Log Form
3.	25	EA-90726-18	Promotional Flyer

TAX: Use MOD Code MA 2% also applicable to Europe

Send To:

Interdepartmental Mail: MR01/T03, Parts Order Processing.

US Mail: Digital Equipment Corporation
10 Forbes Road
Northboro, MA 01532
Attn: Parts Order Processing

LOS (LITERATURE ORDER SYSTEM): You requires an account on the system. Call DTN 234-4308 or DIE 234-4429 to request an account.

The literature warehoused in Northboro is 'owned' by external sponsors associated with various product lines. Sponsors may place quantity limitations or other restrictions on their literature. You should contact the literature controllers for approval of all items which fall within the restrictions. This approval, with the controller's name, must be noted on the order submitted to Order Processing.

Restriction Codes:

1. L = A maximum quantity to be shipped per item has been set by the literature controller.

2. R = Item is currently on hold and the item ordered must be approved by the literature controller (see literature contact list for appropriate person).

3. P = No backorders will be taken on this literature.

INQUIRY SERVICE

To inquire if the desired literature is in stock before ordering, call DTN 234-4325.

1.3 SOFTWARE AND DIAGNOSTIC ORDERING INFORMATION

Use Form: EK-01740-01, Internal Software Order

Mail To:
Interdepartmental Mail: MR0, Software Distribution Center

US Mail: Digital Equipment Corporation
Software Distribution Center
1 Digital Drive
Westminster, MA 01473

Diagnostic Updates: To be placed on automatic distribution for VAX diagnostic updates, contact:

Colorado Springs - DTN 538-5050, or
1-800-525-6570
Europe: locations - Contact ISO

Be prepared to furnish the following information:

Name
Badge Number
Cost Center
Address/Location
Diagnostic Media Type (or part number)

1.4 FIELD SERVICE MICROFICHE LIBRARY

There are four Field Service Microfiche libraries: VAX, PSG, PDP11/PDP8, and VAX. Each library contains information for each product within the group. This information includes hardware manuals, 1052, PM Procedures, diagnostic listings, KIO/XIO information, VMSKB files, wirelists and assembly data. Each library is maintained via quarterly updates and weekly SPEED BULLETIN distributions.

For additional information on these libraries, contact your local branch microfiche librarian or contact:

KSUgP Micropublishing
Information Management and Publishing
Digital Equipment Corporation
30 North Avenue
Burlington, MA 01803

Location code: PTC/B3
DTN: 293-6281
Outside line: (617)273-6281
ENET node: RAINBW::

1.5 RELATED INFORMATION DISTRIBUTION

Being the most complex system Digital has produced yet, the VAX 8600/8650 requires a more complete information distribution system than was in place before.

In addition to using all existing communication channels, we have decided to heavily use SID and the NOTES facility to allow for fast and efficient distribution of system related information.

SID lends itself towards fast, symptom based search of a DATABASE scheduled to contain complete information on currently known VAX8600 related Problems, Fixes, Hints, Revision Information, etc... It will also contain copies of the distribution lists used to distribute system info to allow for timely and accurate update.

Should you feel that your Regional SID database might not contain the most current information towards the problem you are currently working on, then you may use the captive SID account (a FREE node MKNYOU:: Username: SID Password: VENUS.

NOTES will serve as an interactive communications channel between all 8600 interested groups within Digital. The Venus CSSE group will monitor MKNYOU::SYS$NOTES: and work to resolve any upcoming issues.

Both these systems require that you can get access to the ENET, which seems desirable anyways. We assume that all support functions do have access, so everybody should be able to get to the information.

As both systems contain SIGNIFICANT CONFIDENTIAL information, we have to control all surrounding security issues. Accordingly you must not allow Customers any access to information published in there. Also, some entries in SID will only be readable by privileged users (sectional Support).

We encourage any feedback to MONTOR::FOBIEBIUCS

1.6 VAX STUFF PUBLICATION

VAX STUFF is an information newsletter published monthly by Customer Services Systems Engineering (CSSE). The Document contains articles pertinent to all 32-bit VAX systems and peripherals (our MicroVAX through the High end VAXs). VAX STUFF is available on distribution in SPIED BULLETINS under the MISCELLANEOUS section and is also available in hardcopy.

Any Digital Employee (700 and 800 Cost Centers only) wishing to be added/deleted to the VAX Stuff distribution list must submit a memo, INX, or electronic mail to:

```
Julia Jostenchen
OGN1-2/F16
DTN:        276-8950
Outside:    (617)436-8950
INX mail:   OGO
Enet:       IVIVA::JOSTENCHEN
```

Your memo should include the following information:

- your name
- your badge number
- cost center
- location code
- ENET node (if applicable)
- whether or not you wish to be added or deleted from the distribution list

Any articles which you feel should appear in VAX STUFF may be submitted to:

```
VAX STUFF Editorial Office
c/o Julia Costenchen
    OGO1-2/F16
    DTN:        276-8950
    INX CODE:   OGO
    ENET node:  IOIM::JOSENCHEN
```

CHAPTER 2

MODULE UTILIZATION AND BACKPLANE INFORMATION

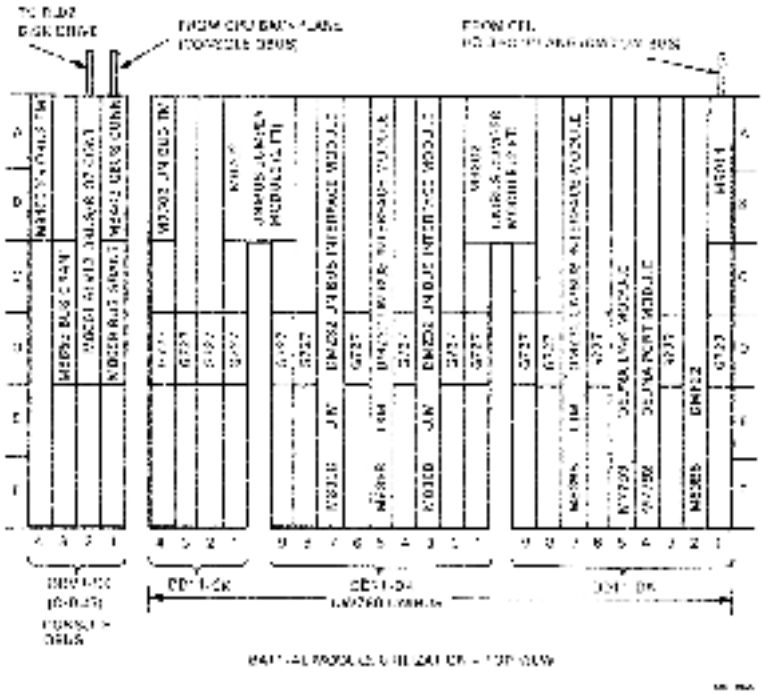


Figure 2-1 BA11-KL Module Utilization as used in VAX 8600/8650 Front-end Cabinet

2.1 VAX 8500/8650 BA11 MODULE UTILIZATION

1. The standard UNIBUS configuration contained in the front and cabinet, BA11 expansion drawer, is as follows:

Qty	Option	Cabus Address	Vector	Nodule(s)
1	DMB32	760340	300	M3196
1	DMB32	760540	310	M3196
		760600	370	
		760640	420	
		760700	430	
1	DZ11A	774510	120	M7792,M7793

2. The standard UNIBUS configuration listed above is included as part of every VAX 8600/8650 system. Due to environmental considerations (air flow and power supply load) this is the maximum allowable configuration on this UNIBUS. DO NOT EXPAND THE CONFIGURATION BEYOND THE STANDARD.

3. For additional configuration information, refer to Chapter 7, System Configuration Guidelines.

4. All unused UNIBUS slots have a G727 Grant Continuity module installed in slot D.

5. All unused UNIBUS slots have an NPR jumper wired on the backplane from pin CA1 - CB1. Note that in the case of the UNIBUS, the NPR jumper is removed from slot 4 (M7792) but installed in slot 5 (M7793).

6. Note that this BA11 expansion drawer contains two separate buses: the Qbus for the RL subsystem and a UNIBUS for the first DW780.

7. For power distribution information, refer to Chapter 3, H7642-CA Power Distribution.

8. For Qbus cabling information, refer to Chapter 9, Qbus Interconnect.

9. For DW780/UNIBUS cabling information refer to Chapter 10, SBI NEXUS AND INTERNAL OPTIONS and to the VAX 8600/8650 Installation Manual.

10. For UNIBUS Option information (DZ11A, DMF32, and DMZ32) refer to the VAX 8600/8650 Installation Manual and to the appropriate DEVICE manual (see Chapter 1 for a list of related documents).

2.2 VAX 8600/8650 MEMORY ARRAY MODULE CONFIGURATIONS

The following guidelines should be used to configure the VAX 8600/8650 Memory Subsystem. The memory modules used include the following:

Option Designation	Module Number	Revision Information	Module Capacity	Used On 8600	Used On 8650	Note
MS86-A	LQ226	any	4 MBytes	xx	xx	
MS86-B	LQ200	C1 and lower	6 MBytes	xx		
MS86-D	LQ200	D1 and higher	4 MBytes	xx	xx	
MS86-C	LQ225	any	16 MBytes	xx	xx	2
MS86-D	LQ235	any	64 MBytes	xx	xx	3
N/A	LQ200		N/A	xx	xx	4

NOTES

1. With the exception of the LQ200 (rev C1 and lower) module all modules can be used interchangeably in either a VAX 8600 or VAX 8650 System.

2. The MS86-C is comprised of an LQ225 mother board and 8 daughter boards (PN 54-16500-AA or 54-16500-BA). Each LQ225 plugs into one slot on the backplane; however, each LQ225 will occupy two slots.

3. The MS86-D is comprised of an LQ235 mother board and 4 daughter boards (PN 54-17052-AA). Each LQ235 plugs into one slot on the backplane; however, each LQ235 will occupy two slots.

4. The LQ200 Memory Backplane Module is used to balance power regulator loading.

Configuration Rules:

1. Populate the Memory Backplane in ascending order beginning with slot 1.

2. Install larger capacity memory boards first, installing 64 MByte modules and 16 MByte modules before 4 MByte Modules.

3. Follow the "Used On" column in the table above.

4. Install LQ200 Memory Load Modules in slots 8 and 9 unless these slots are occupied by an Array Module.

5. Always install an LQ232 Memory Terminator Module in slot 9 of the Memory Backplane.

2.3 CARIKEI CAGE MODULE UTILISATION

Figure 2-2 CEU Cabinet Cage Module Utilization

2.4 CPU, ABUS, AND MEMORY BACKPLANE COMPONENTS AND PIN ASSIGNMENTS

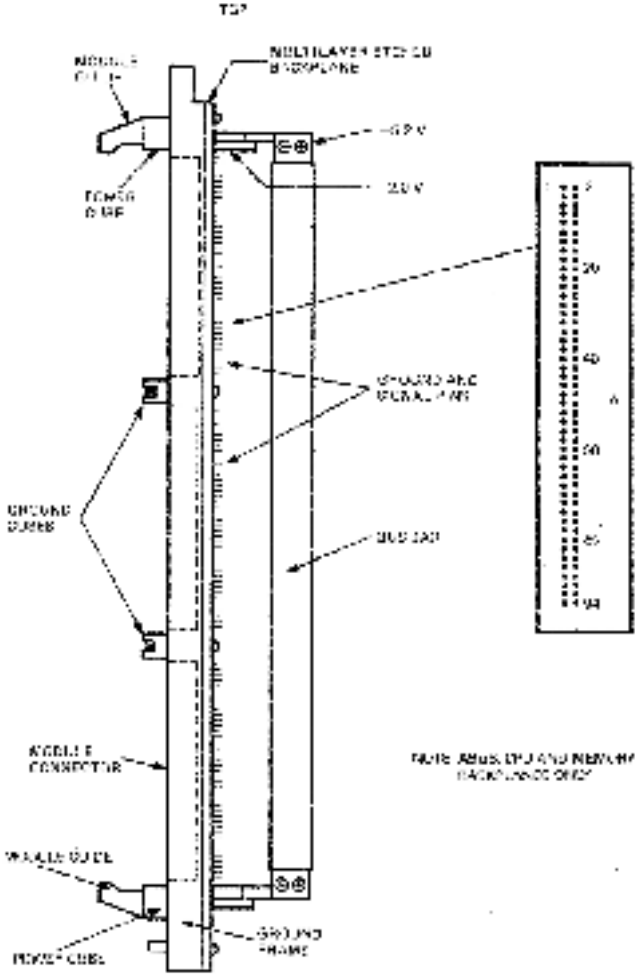


Figure 2-3 CPU, ABUS, and Memory Backplane Components - Side View

re 2-6 I/O Backplane Pin Numbering Scar

2.0 CABINET CAGE BACKPLANE - REAR VIEW

Note: See Backplane Interconnect Chart for a list of cables and interconnections.

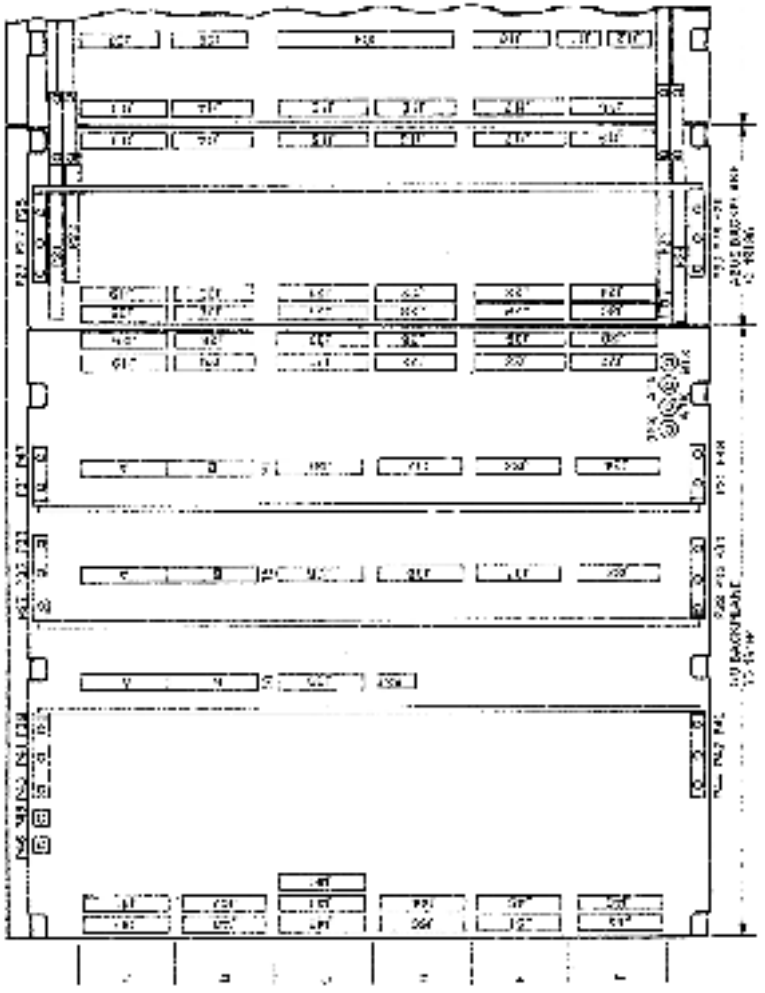


Figure 2-3 Cabinet Cage Backplane (Sheet 1 of 2)

Figure 2-5 Cabinet Cage Wireline (Sheet 2 of 2)

2.7 BACKPLANE INTERCONNECTIONS

FROM		TO				
Pin	CPU	Pin	Pin	Part No.	Qty	Description
P1	J21-A15	VME	J21-A16	12-03282-01	6	Cable assembly, 50 ohm, 2.6 inches (6.6 cm), Dual
P2	J27-C2	P2 Slot 20A	J1-C2	12-03282-12	6	Coaxial Cable
P3	P3	-	-	-	-	Slot Jumper
P4	J13	Alarms Connector Backpanel	P3	9700-100	1	Ribbon Assembly (Used with Interface Card 2)
P5	J11	P15 Backpanel	J11-A16	12-10005-01	1	11TV Video Trunk
P6	J13 J16	P16 Backpanel	J13-A16 J16-A16	12-10005-02 12-10005-03	1 4	Cable Assembly, Coaxial-502 Cable Assembly, Coaxial-503 Cable assembly, 75 ohm, 2.6 inches (6.6 cm)
P7 P8	J14-J15 J25-J20	J11 J11	J14-A16 J25-A16	12-03282-13 12-03282-11	4 4	Cable assembly, 75 ohm, 5.6 inches (14.1 cm) Cable assembly, 75 ohm, 5.6 inches (14.1 cm)
P9 P10	J21 J4	P9214 to J4 on I-Bus	J1-J5	12-03282-14	2	Configuration Jumper, LI750 Unused cable as jumper (Part No. 12-03282-14)
P11	J25, J3	P9214 Any Unused Slot	J1-A	12-03282-14	2	Configuration Jumper (Part No. 12-03282-14) Unused cable as jumper (Part No. 12-03282-14)
P12 P13	J20 J21, J26	- P9214 Any Unused Slot	- J1-A	- 12-03282-14	- 4	Configuration Jumper (Part No. 12-03282-14) Unused cable as jumper (Part No. 12-03282-14)
P14	J27, J26	P1202 to J21 on Backplane	J1-A16	12-03282-14	2	LSI I Interface Cables
P15	J27, J25	P1202 to J21 on Backplane	J1-A16	12-03282-14	2	LSI I Interface Cables
P16 P17	J11, J3 J25, J3 J26, J3 J27, J3	P9214 Any Unused Slot	J25, J18	70-31403-01	1	Terminal Board X Adapter Cable Set Part No. 99214 Backpanel
P18	J25, J3 J26, J3 J27, J3	-	-	-	-	Jumper, Unused IBU Interface
P19 P20	J25, J3 J26, J3 J27, J3	- P2 Extension Slot	- J1	70-10983-01	1	Bridge Module Interface Cable Ribbon Cable Assembly
P21	J3	Internal Slots Unused	-	-	-	Connector on the rear internal slot. Caps

NOTE

Refer to Schematic 12-31919-10 and 11-10710-10 in Volume 1 of the Multibus System Manual for a description of RAMB interconnections.

CHAPTER 3

POWER

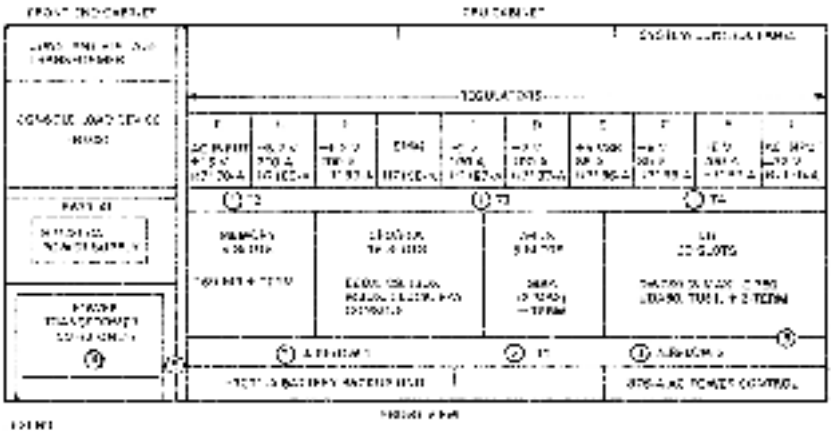


Figure 3-1 Dover System Components

3.1 POWER SYSTEM COMPONENT DESCRIPTION

Part Number	Description
12-19526-01	Temperature Sensor.
12-22805-01	Air Flow Sensor (with red LED).
17-00932-01	Cable Assembly (for connecting Drawer 626 Line Monitor to 626 System).
5706A	System Power Controller. Provides switched and unswitched ac power distribution as well as DDC power bus interconnections.
5714C-03A	Front and cabinet 8All-AI expansion drawer power supply. Provides the necessary dc power for all backplanes within this expansion drawer.

H7170-A	MPS ac Input Modules which provide 300 Vdc as input to the low voltage regulators. Also provide low power ± 12 and ± 15 volts.
H7180-A	MPS +5 V and -5.2 Vdc Regulators (200 Amps).
H7186-A	MPS +5 Vdc Regulator (85 Amps).
H7187-A	MPS -2 Vdc Regulator (100 Amps).
H7130-A	Environmental Monitor Module (EMM). Provides control of the MPS and monitors the environment of the CPU. Communicates with the console via an asynchronous interface.
H7251-A	Battery Back-up Unit which supplies +5 V for the IOX block and 255 Vdc (@ 3.8 Amps) for 10 minutes minimum upon a power failure.
3023959-01	Constant Voltage Transformer (CVT) 60 Hz-Rev B1
3023959-02	Constant Voltage Transformer (CVT) 50 Hz-Rev B1

3.2 ENVIRONMENTAL SENSOR DESCRIPTIONS (AND LIMITS)

There are a number of sensors located within the system which monitor various aspects of the system environment. These monitoring devices work in conjunction with the hardware and software to ensure that abnormal environmental conditions are reported and, in some cases, power down the system to prevent pending damage from occurring.

The following sections describe the various sensors used, how they operate and what their thresholds are set at.

3.2.1 Air Temperature Sensors

There are four air temperature sensors installed in the VAX 8600 system CPU cabinet. These sensors are used to monitor local conditions of the CPU cabinet and module card cage.

3.2.1.1 Ambient Air Input Temperature Sensor (T1) - this sensor monitors the CPU cabinet input temperature. The programmable thresholds are currently set at 33 degrees C (yellow zone) and 35 degrees C (red zone).

If the sensor detects that the temperature has dropped below 16 degrees, an informative message will be printed on the console terminal. Since this is a non-fatal condition, a pending shutdown will NOT be initiated.

3.2.1.2 Module Card Cage Exhaust Temperature Sensors (T2, T3, T4) - These sensors measure the exhaust temperature at three separate locations above the CPU cabinet module card cage. The programmable thresholds are currently set at 48 degrees C (yellow zone) and 55 degrees C (red zone).

If the sensor(s) detects that the temperature has dropped below 10 degrees, an information message will be printed on the console terminal. Since this is a non-fatal condition, a pending shutdown will NOT be initiated.

3.2.1.3 Air Temperature Increase Across Modules (All Sensors) - This measurement is achieved by monitoring the difference (delta) between the card cage input temperature sensor (T1) and each of the exhaust temperature sensors. This results in three measurements as follows:

"Delta Temperature T1-T2"
"Delta Temperature T1-T3"
"Delta Temperature T1-T4"

The programmable thresholds are currently set at 12 degrees C (yellow zone) and 20 degrees C (red zone).

3.2.1.4 Critical Cabinet Temperature Sensor (T3) - In addition to providing a reading for the temperature measurements outlined previously, this sensor also feeds a hardware circuit in the EMS which forces a "D/A" UPS system power down when the temperature exceeds 60 degrees C.

3.2.2 Air Flow Sensors

There are two air flow sensors present in the CPU cabinet which are located directly below the module card cage. The outputs of these sensors feed the CPM which in turn will report when the air flow drops below a specific amount, allow a specified amount of time for an operator to correct the situation and, if not remedied, power down the system.

3.2.3 Ground Current Sensor

A ground current sensor is located in the front end cabinet which measures ground current on the main ground conductor where it enters the cabinet.

There is no "limit" set for this measurement. Rather, the reading will only be taken when requested by the operator (via the "SHOW CONSEN" console command). Field Service should use this sensor to verify that the ground current is within the specified tolerance (100 milliamps) and that the current does not change drastically over a period of time.

3.2.6 50-Hertz Transformer Temperature Sensor (Front End Cabinet)

There is a temperature sensor installed on the power transformer used in 50 Hertz systems. This temperature sensor is monitored continuously and will cause an Emergency Power Off if the temperature rises above 175 degrees Celsius (+ 5 degrees).

3.2.7 CPU Cabinet Filter Sensor

A mechanical switch located at the right rear of the mobile card cage in the CPU cabinet detects the presence (or absence) of the cabinet filter. If the filter is not installed, the system will not power up. Furthermore, if the filter is removed while the system is up and running, the system will power down.

3.3 ENVIRONMENTAL MONITORING AND EMERGENCY POWER OFF CONDITIONS

The System has a number of environmental monitoring schemes that are used to monitor and report specific ambient conditions and, under certain circumstances, power down the system due to pending critical environmental conditions. All the schemes fall into one of the following categories.

3.3.1 EPO Hardware Monitored and Controlled

When these monitored conditions reach a specified threshold, they immediately force an emergency power off of the entire system. The main circuit breaker on the 875-A power controller is tripped and the four voltage (indicators) on the EPO, indicate the source problem.

Refer to "8750-A EPO POWER - VISUAL INDICATORS" for more information on the EPO Total DC module.

Refer to "Environmental Sensors Descriptions (and Limits)" for more information on the sensors.

3.3.2 EPO Software Monitored and Controlled

This EPO mode of operation is referred to as ASD (Auto Shutdown). Refer to "ENVIRONMENTAL SENSOR DESCRIPTION (AND LIMITS)" for more information on the conditions monitored and thresholds.

If a monitored condition reaches a specified threshold, the console will be notified to warn the operator of a pending environmental fault. Various indicator LEDs will also be lit. If the condition is not restored within a specific amount of time, or the condition reaches a second higher threshold, an emergency power off of the entire system is initiated and the main circuit breaker on the 875-A power controller is tripped.

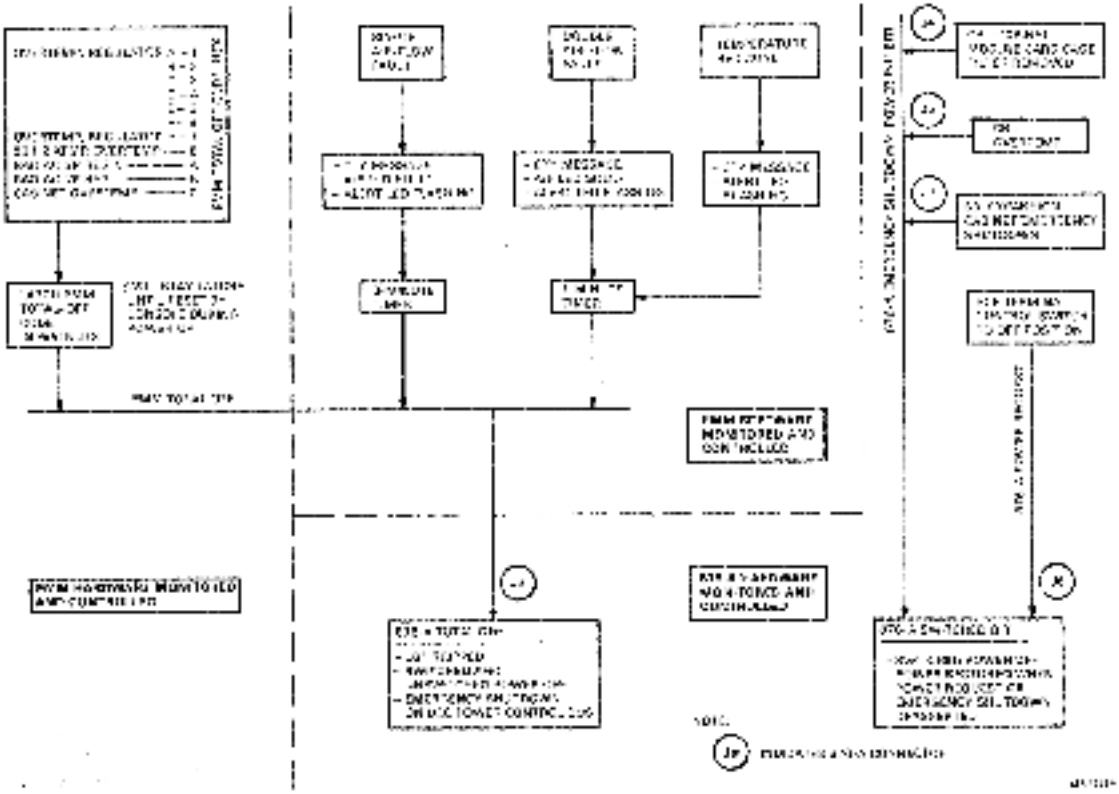


Figure 5-2 Power Shutdown Sequence

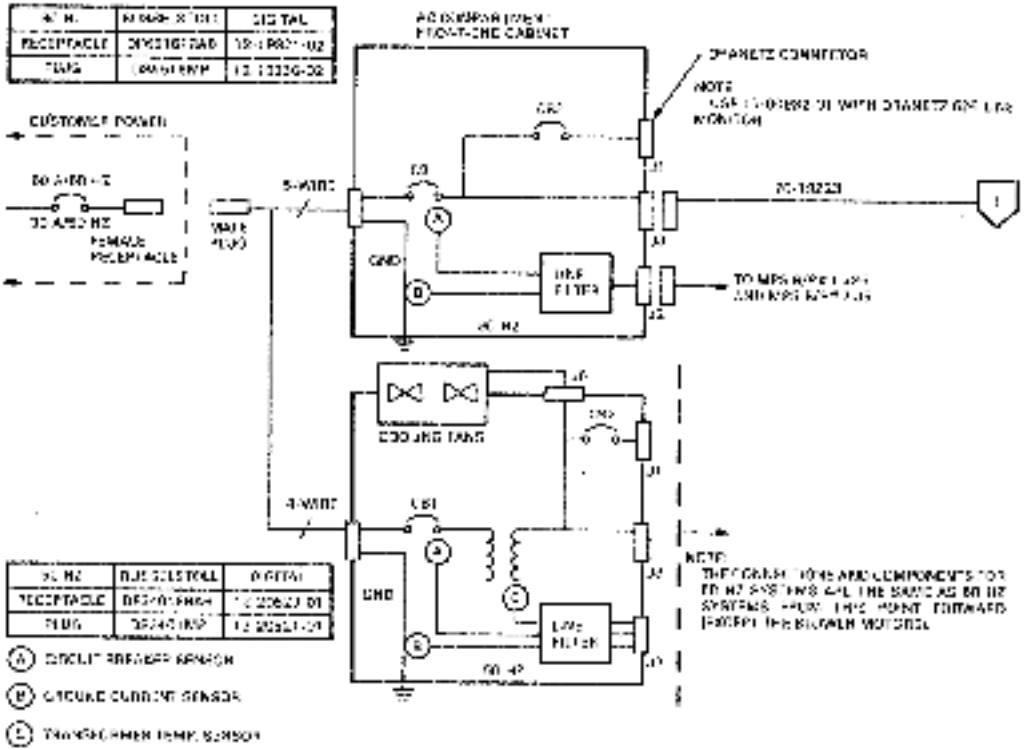


Figure 2-2 Power System Interconnect Diagram

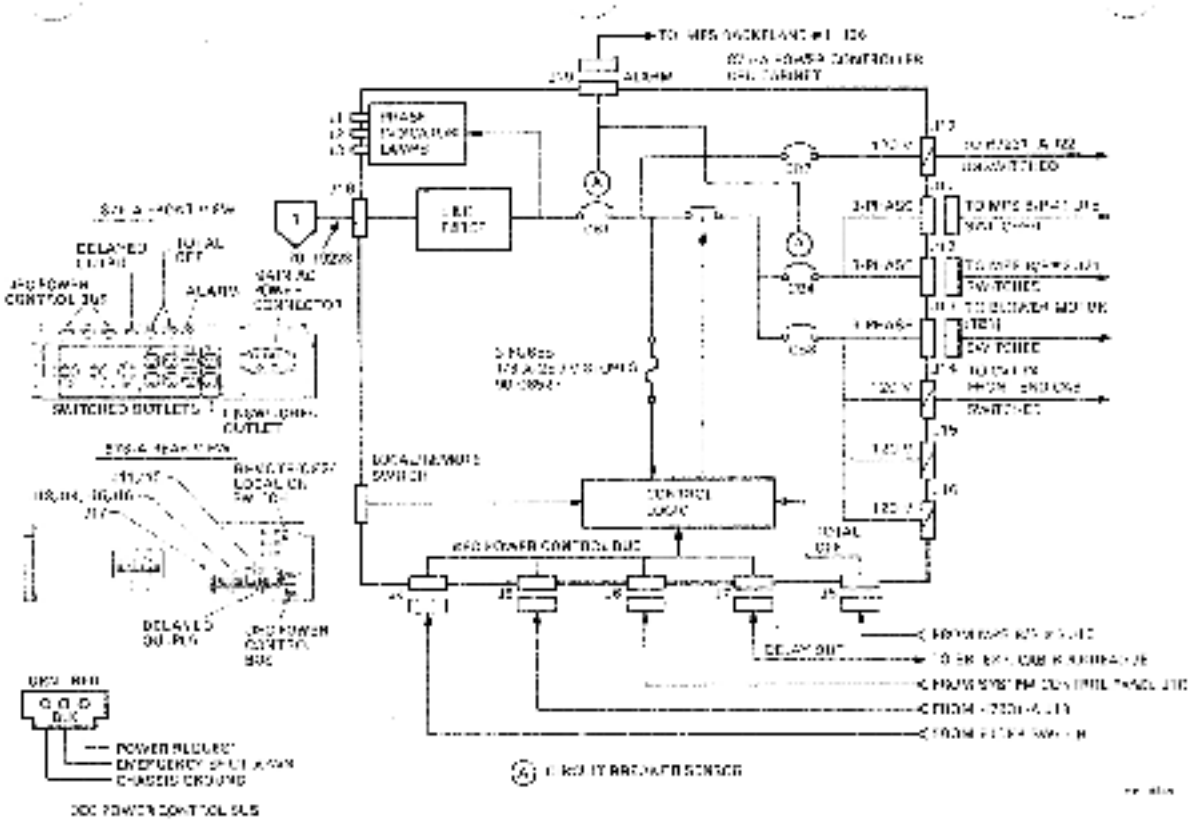


Figure 3-4 Power System Interconnect Diagram

3.3.3 EMM Software Monitored and Console Software Controlled

These conditions, under Console Software control, are monitored and reported to the Console Software. These conditions will not cause an automatic power off, rather they will be reported via an error message in the Console terminal.

This EMM mode of operation is referred to as the "Environmental Monitor" mode. Refer to "ENVIRONMENTAL MONITOR DESCRIPTION (AND LIMITS)" for more information on the conditions monitored and thresholds.

3.3.4 H764-A Power Controller Monitored and Controlled

These conditions, monitored by various sensors, are intended to prevent system power-up under unsafe conditions as well as forcing EMERGENCY system shutdown when an unsafe condition is detected.

NOTE

With the exception of "EMM Software Monitored and Console Software Controlled", all these schemes may result in a system power down situation. To help illustrate the interaction of these systems, refer to the "Power Shutdown Sequence" flowchart. Also, the section titled "Environmental Monitoring - Software Control" gives additional information on the software timing of the air flow and temperature faults, as well as a description of the warning messages.

3.4 REFERENCE DOCUMENTATION

DEC STD 123 (Power Control Bus Standard)	EL-00123-00
ZASE EMM Technical Description	EK-ZASEV-TD
3459 System Power Technical Description	EK-KA46P-TD
VAX 6000/6000 Field Service Maintenance Print Set	MP-01716-00
VAX-11/780 VAXBI System Technical Description	EK-BI780-TD

3.5 SYSTEM ENVIRONMENTAL MONITORING - SOFTWARE CONTROL

There are a number of environmental parameters monitored by the EMM software and in some cases controlled by the console software. When a change of status in any of these parameters is detected, the action taken will depend on whether or not the console is in CTO mode, and whether or not the change is fatal.

Upon detection of a change of status:

- If the console is in CTO mode, a descriptive error message is printed on the console terminal describing the status change.

- If the console is in PIO mode, the following "generic" console message will be printed:

  "%SYSTEM, Environmental Alert - Environmental Monitor has detected an alert condition. Please check the error log."

  As outlined in this message, it will be necessary to consult the error log to determine the source of the alert condition.

- If the status change is not corrected within a specified amount of time, a total system power down is initiated, and a software timer (Automatic Shutdown timer, or ASD) begins. If the situation is not corrected within the amount of time specified by the timer, then the system will be shutdown by the KPM. Furthermore, if the console is in PIO mode, the "generic" message outlined above will be appended by the following:

  "Total system power shutdown pending if condition is not corrected."

- If the status change is not catastrophic in nature, then the only action taken by the software is to report the change via the messages outlined above.

Each of these conditions, with the action taken, is outlined below.

3.5.1 Temperature Sensor (or Delta Temperature) Enters Yellow Zone

When a monitored temperature parameter enters the yellow zone:

- a warning message is typed on the console terminal.
- the ALERT LED will be lit SOLID.

As this is considered a NON-FATAL situation, no other action is taken.

3.5.2 Temperature Sensor (or Delta Temperature) Enters Red Zone

When a monitored temperature parameter enters the RED zone:

- A warning message is typed on the console terminal.
- the ALERT LED flashes
- the Automatic Shutdown Timer is set to one minute.

Since the ASD timer has been set to one minute, a pending shutdown has begun. If the RED ZONE temperature condition is not rectified within one minute, the KPM initiates a 'FATAL-ON' condition which trips the 870-A main circuit breaker.

3.5.3 Single Air-flow Fault

When an AIR-FLOW fault is detected, the following occurs:

- a warning message is typed on the console terminal.
- the AIR-FLOW sensor LED lights solid.
- the ALERT LED on the BCPI flashes.
- the Automatic shutdown timer is set to three minutes

Since the AST timer has been set to three minutes, a pending shutdown has begun. If the air flow fault condition is not rectified within three minutes, the EMM initiates a IOTAL-OFF condition which will trip the 870-A main circuit breaker.

3.5.4 Double Airflow Fault

When a second AIR-FLOW fault is detected, the following occurs:

- a warning message is typed on the console terminal.
- the second AIR-FLOW sensor LED lights solid.
- the ALERT LED continues to flash.
- the Automatic shutdown timer is set to one minute.

Since the AST timer has been set to one minute, a pending shutdown has begun. If the double air flow fault condition is not rectified within one minute, the EMM initiates a TOTAL-OFF condition which will trip the 870-A main circuit breaker.

3.5.5 MPS Status Change

When the EMM and/or console software detect a change in the status of the MPS system a message is printed on the console terminal outlining the status.

3.6 MPS POWER DISTRIBUTION AND COLOR CODES

Reg	Part	Voltages	Load (Backplane and/or Modules)
A	H7186-A	+5 V	EMM (H7186-A), CSL (LO201), I/O Backplane, Clock Module (LO217/TO211)
B	H7186-A	+5 V	Memory Backplane (this regulator has EMM)
C	H7186-A	-5 V	ABus Backplane, Memory Backplane
D	H7187-A	-2 V	CPU and ABus Backplanes (Note 1)
E	H7187-A	-2 V	CPU and Memory Backplanes (Note 1)
F	H7180-A	-5.2 V	CPU and Memory Backplanes (Note 2)
H	H7180-A	-5.2 V	CPU, ABus and I/O Backplanes (Note 2)
K	H7170-A	± 15 V + 100 V	I/O Backplane Regulators K and H
L	H7170-A	± 12 V + 100 V	XMM (H7186-A), CLK (LO217 for VAX 8600, LO231 for VAX 8650), CSL (LO201) Regulators A through F
BBU	H7231-A	+5 V	TOY (LO201)

NOTES

1. H7187-A Regulators D and K outputs (+2 9) are tied together on the backplane.

2. H7189-A Regulators F and H outputs (-5.2 V) are tied together on the backplane.

3. The H7180-A Regulator is used for both the -5.2 V and +5.0 V supplies. The actual regulator is the same in all cases with the output voltage being "selected" by insertion into the backplane (i.e., if plugged into slot A it will provide +5 V, whereas if plugged into slots F or F it will provide -5.2 V).

4. The H7170-A Regulator provides the 300 Vdc bus input to the other regulators as well as low power + 12 V and + 15 V. These two regulators are interchangeable; selection for the 12/15 V output is made by a jumper on the MPS backplane.

3.7 MPS POWER DISTRIBUTION DESIGN CODE CHART

Color	Voltage
BLACK	GROUND
RED	+ 5 V
GRAY	+ 2 V
VIOLET	- 5.2 V
GREEN	+ 15 V
YELLOW	+ 12 V
ORANGE	+ 12 V
BROWN	- 12 V

3.8 HOW TO MARGIN MPS REGULATORS

It is possible to margin some of the MPS dc voltage regulators using a console command. The margining is actually controlled by the BMB. Of the voltage outputs from the MPS, the +5, -5.2 and -2 volt outputs are marginable, the + 15 and + 12 volt outputs are not. Also, as the -5.2 and -2 volt outputs are tied together on the backplane, these regulators must be margined in pairs; the +5 volt regulators can be margined separately.

To margin an MPS regulator, use the SET MARGIN command. For a detailed description of this command refer to the Console Software Specification. A brief description of the margining procedure follows.

CPU BACKPLANE

VOLTAGE	SILKSCREEN	DIP PIN	REGULATOR
GROUND			
+5 V			
+5 V (LV)			
+12 V			
-12 V			
-5.2 V			

MEMORY BACKPLANE

VOLTAGE	SILKSCREEN	DIP PIN	REGULATOR
GROUND			
+5 V			
+5 V (BU)			
+12 V			
-5.2 V			

ABUS BACKPLANE

VOLTAGE	SILKSCREEN	DIP PIN	REGULATOR
GROUND			
+5 V			
+12 V			
-5.2 V			
ENWS DBIA A, LOW			
ENWD DBIA D, LOW			

I/O BACKPLANE

GROUND			
+5 V			
+12 V			
-12 V			
-15 V			
SLOT0 ADDR, LOW			
SLOT1 ADDR, LOW			

NOTE:
THESE PART LISTINGS ARE FOR REFERENCE ONLY AND DO NOT NECESSARILY END CHECK ALL MANUFACTURER'S DATA BOOKS FOR PRECISE PINOUTS FOR APPROPRIATE MPU. SEE TOP FOR A LIST OF ALL CONNECTIONS.

Figure 3-5 MPS Voltage Measurements

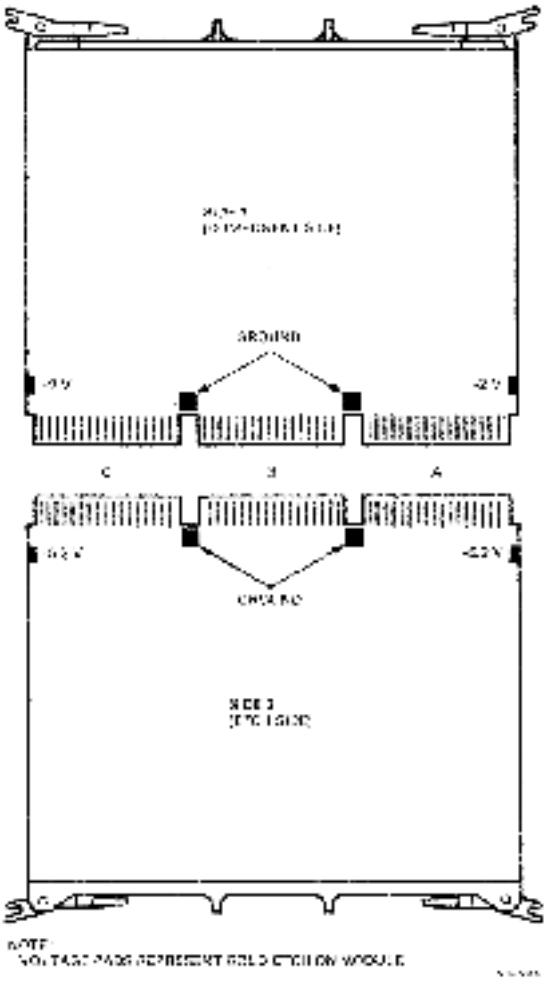


Figure 3-6 Module Power Connections

3.8.1 MPS Regulator Margining Procedure

1. Determine which regulator/voltages you wish to margin. Referring to the regulator location below, select one or more of the following:

Regulators H and I	-5.2 V
Regulators E and D	-2 V
Regulator C	+5 V
Regulator B	+5 V
Regulator A	+5 V

Reg	I	H	F	FAN	E	D	B	C	A	J
Vlt	[1.5 V	-5.2 V	-5.2 V		-2 V	-2 V	+5 V	+5 V	+5 V	+12 V

2. From CONSOLE I/O mode, enter one of the following commands to the VAX 9000 Console:

>>>SET MARGIN HIGH reg#	Increases the voltage, for the selected regulator(s), by 5%.
>>>SET MARGIN LOW reg#	Decreases the voltage, for the selected regulator(s), by 5%
>>>SET MARGIN NORMAL reg#	Returns the voltage, for the selected regulator(s), to normal.

NOTE

Reg# refers to the selected regulator(s) (e.g., A, H, I, HK, FN, or ALL) and if omitted, all regulators (A through H) will be selected.

3. To verify/measure the margined voltage, use the SHOW POWER command (see example in section on MPS Power and Sensor Reading) or measure the voltage on the appropriate backplane (see MPS Voltage Measurements).

4. When completed, be sure to return the voltages to normal using either the SET MARGIN NORMAL command or the INIT/TOKBR command.

3.9 H7179-A POWER SUPPLY - VISUAL INDICATORS

LED	CONDITIONS
BUS OK	GREEN LED. Indicates when the voltage is within 185 to 215 Vdc. The minimum voltage required for MPS regulator operation is 165 Vdc. The power now supplied by H7179-A units E and I.
	The Bus OK LED does not light when the H7179 units do not receive ac input power, and blinks when the ac power is less than specified (below 156 V RMS line to line based on a 208 Vac input).
MODULE OK	GREEN LED. Indicates when the H7179-A unit is supplying voltages that are within proper regulation range, and no other faults are present.

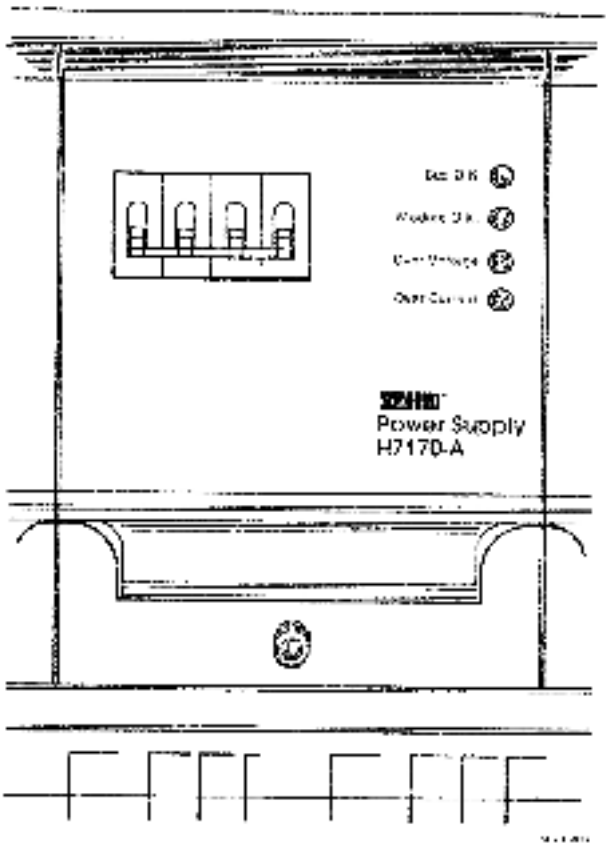


Figure 3-7 H7170-A Power Supply - Visual Indicators

OVER VOLTAGE RED LED. Indicates when any or all auxiliary voltages (in an H7170) crowbar. Input power must be removed to reset this signal.

Overvoltage trip operation is as follows:

+12 V output:	+12.8 to +13.6 Vdc
-12 V output:	-12.8 to -13.6 Vdc
+15 V output:	+16.2 to +16.9 Vdc
-15 V output:	-16.2 to -16.7 Vdc

OVER CURRENT YELLOW LED. Blinks when output current is above 2 Amps at either (or both) the 12 volt and 15 volt output.

3.10 H7180-A, H7186-A, AND H7187-A POWER SUPPLY VISUAL INDICATORS

LED	DESCRIPTION
MODULE OK	GREEN LED. Indicates that the regulator output voltage is within the regulation range and that no faults are present.

Regulator		Regulation Range
H7180-A	[A]	−5.0 V ± .005 Vdc
H7186-A	[K,F]	−5.2 V ± .005 Vdc
H7185-A	[D,C]	−5.0 V ± .075 Vdc
H7187-A	[H,J]	−2.0 V ± .015 Vdc

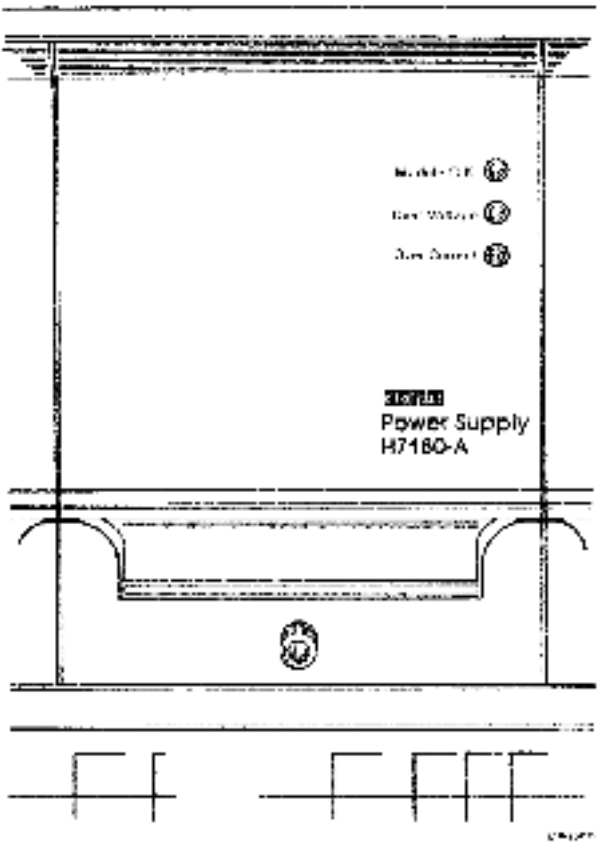


Figure 3-8 H7180-A Power Supply

OVER VOLTAGE

RED LED. Indicates that an overvoltage condition has occurred. This condition will latch and the voltage output will be crowbarred.

Regulator	Overvoltage limit
H7180-A (A)	+6.0 V
H7180-A (F,H)	+6.0 V
H7186-A (B,C)	+6.3 V
H7187-A (D,E)	+2.0 V

OVER CURRENT

YELLOW LED. Blinking LED indicates that the regulator output current is above the maximum rating.

Regulator	Maximum output current
H7180-A (A)	200 A
H7180-A (F,H)	200 A
H7186-A (B,C)	85 A
H7187-A (D,E)	100 A

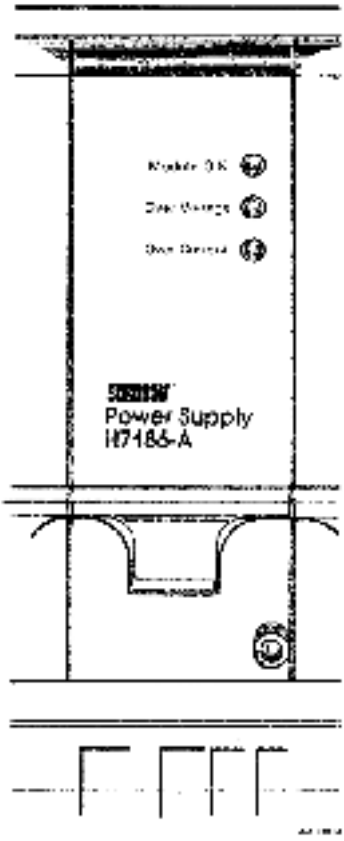


Figure 3-9 H7180-A (H7187-A) Power Supply

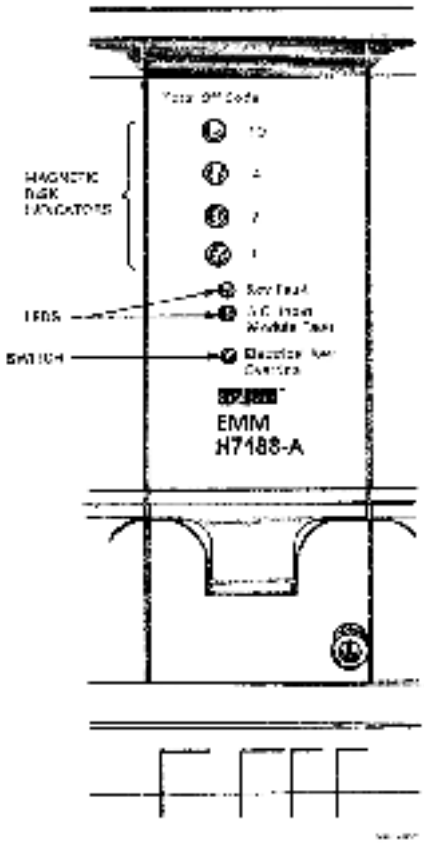


Figure 3-10 H7188-A EMM Module Visual Indications

3.11 TOTAL OFF CODES - FAULT CONDITIONS

Total Off Code					
10	8	4	2	1	Octal	Condition
0	0	0	0		0	BMM program turns off ac power in response to a console initiated command (normal power off condition)
0	0	0	1		1	Overtemperature, Regulator A
0	0	1	0		2	Overtemperature, Regulator B
0	0	1	1		3	Overtemperature, Regulator C
0	1	0	0		4	Overtemperature, Regulator D
0	1	0	1		5	Overtemperature, Regulator E
0	1	1	0		6	Overtemperature, Regulator F
0	1	1	1		7	Overtemperature, Regulator G
1	0	0	0		10	Transformer OT (50 Hz only)
1	0	0	1		11	Overtemperature, Spare D (unused)
1	0	1	0		12	Bad ac input, Module K
1	0	1	1		13	Bad ac input, Module L
1	1	0	0		14	Cabinet Overtemperature, Sensor T3*
1	1	0	1		15	Not used
1	1	1	0		16	Overtemperature, Spare C (unused)
1	1	1	1		17	BMM cannot communicate with the Console †

3.12 BMM LEDS AND SWITCHES

LED or Switch	Indication/Function
Key Fault	Red LED. Indicates a missing or misplaced module(s) in memory, CPU, or APUs on backplanes of the VAX 8600/8650 System cabinet card cage.
AC Input Module Fault	Indicates an input error is detected during power up. BPS modules K and/or L are not OK.
Electrical Key Override	This switch will override a SERIAL KEY FAULT, allowing the power-up sequence to continue.

---

* Temperature sensor T3 (below the BMM module) monitors the outlet temperature from the module cage and the inlet temperature to the BMM. In addition to providing temperature data to the BMM/Console (for yellow and red zone temperature monitoring), T3 also feeds a hardware circuit that shuts down the system when the temperature exceeds 50 degrees C. This hardware generated shutdown provides a failsafe circuit should the software monitoring cabinet temperatures fail.

† For this condition, the magnetic disk indicators will flash alternately all on, and then all off.

3.13 MPS POWER AND SENSOR READING

Field	Description
Input	Indicates what parameter/unit is being measured (e.g., temperature or regulator).
Measured	The actual value read. Will be a voltage, current or temperature (in degrees C).
Margin	Viewed only for the marginable regulators and will read:

NORMAL - Voltage is not margined
HIGH - Voltage has been margined high, + 5%
LOW - Voltage has been margined low, - 5%

>>>SHOW POWER

Input	Measured	Margin	Status	Input	Measured	Status
Regl A	+3.02 V	NORMAL	OK	Inlet		
Regl B	+3.02 V	NORMAL	OK	Temp1	22 Deg C	OK
Regl C	+5.02 V	NORMAL	OK			
Regl D	-1.98 V	NORMAL	OK	Outlet		
Regl E	-1.99 V	NORMAL	OK	Temp2	24 Deg C	OK
Regl F	-5.1 V	NORMAL	OK			
Regl H	-5.10 V	NORMAL	OK	Outlet		
Regl K	+15.16 V		OK	Temp3	22 Deg C	OK
	-14.92 V		OK			
Regl L	+12.19 V		OK	Outlet		
	-12.05 V		OK	Temp4	24 Deg C	OK
Bar Cur	98 Milliamps					

Status - Overall status of the unit being measured and will read:

- Normal status, all OK

3.14 MPS STATUS MESSAGES

When the Console is notified of an MPS status change it will appear a message on the Console Terminal. Refer to Chapter 4 of the System Fault Isolation Manual for a description of MPS Status Messages.

3.15 HOW TO ENABLE/DISABLE BATTERY BACKUP

The BBU Unit can be enabled/disabled via console commands. To enable the BBU, enter the following command:

```
>>>SET FLAG BBU ON
```

And to disable the BBU, enter the following command:

```
>>>SET FLAG BBU OFF
```

To determine the status of the BBU (whether or not it is enabled) use the SHOW POWER or SHOW FLAGS command. Refer to the section on MPS Power and Sensor Reading.

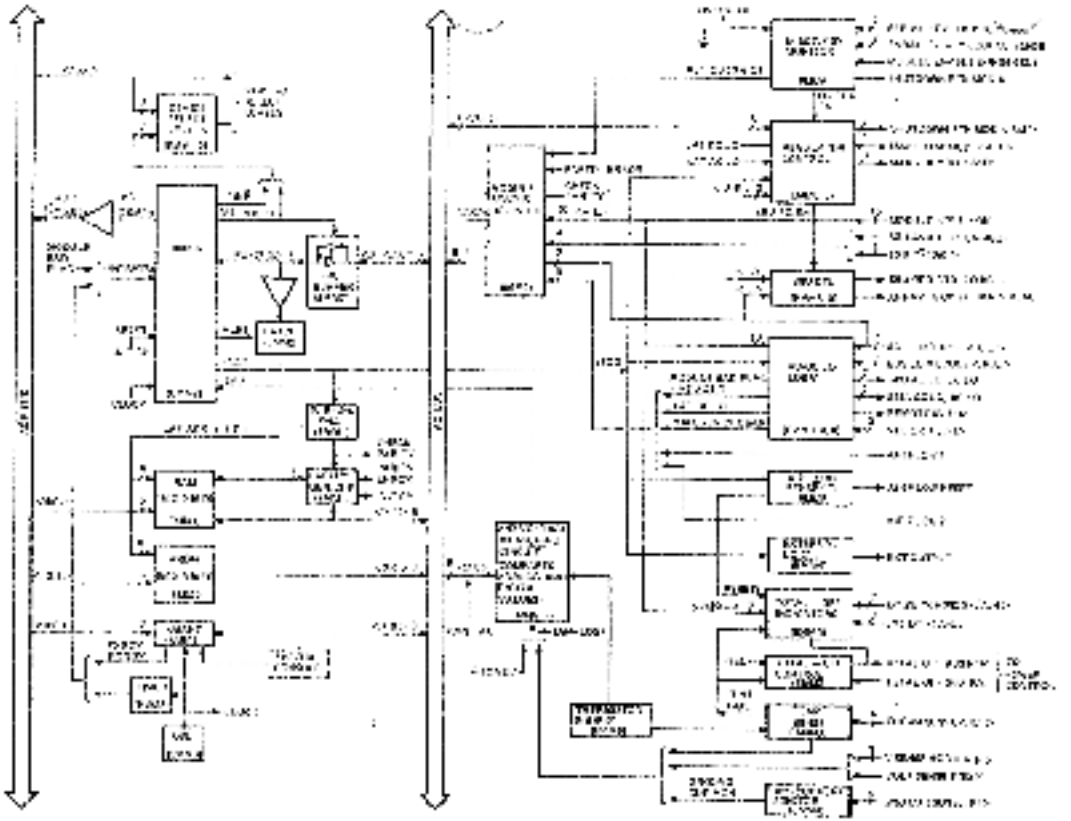


Figure 3-11 EMM Module - Detailed Block Diagram

3.16 HOW TO DETERMINE THE BATTERY CHARGE STATUS OF THE BBU

There is a red LED located on UPS backplane 2 (labeled J2) which indicates the various status of the BBU charging circuit and is read thus:

D2 Indicator	Status
OFF	BBU is either turned off or broken.
ON	BBU is fully charged.
FLASHING (once per sec)	BBU Batteries are charging.
FLASHING (10 times per sec)	BBU is supplying power to memory.

3.17 B7140-CA POWER DISTRIBUTION

The B7140-CA supplies power to all backplanes within the BA11-K OKIBUS expansion drawer in the front end cabinet. There is a total of 4 backplanes within the expansion drawer: two 9-slot DD11-DK and one 4-slot DD11-CK backplanes for the UNIBUS on SW785-D ; and 6-slot DSV11-CK backplane for the Console QBus. The power connections for each of these backplanes is made via Mate-N-Lok connections to the B7140-CA. The interconnections are shown in the following figure.

The table below lists the color codes used on each of these power harnesses. The B7140 Power Distribution/Interconnection Tables list the interconnection and backplane test points where the voltages can be measured for each of the three different backplanes.

NOTE

> The B7140-CA is an FRU as a complete unit. Voltage adjustments are done at the factory and should not be attempted in the field.

3.17.1 B7140 Power Distribution Color Codes

Color	Voltage
BLACK	GROUND
RED	+5 V
GRAY	+15 V
WHITE	+15 V
BLUE	-15 V
GREEN	-15 V
BROWN	-5 V or L7C
ORANGE	+20 V
VIOLET	DC Low
YELLOW	AC Low

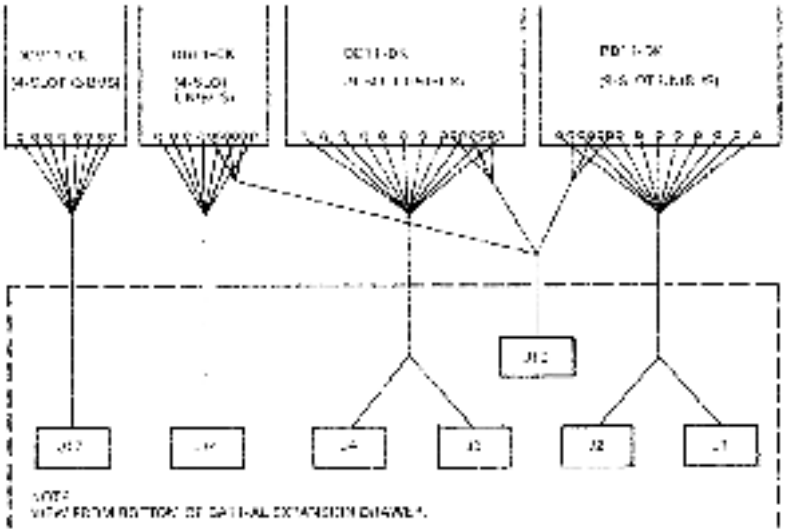


Figure 3-12 a7140-CA Power Interconnection Diagram

3.17.2 37140 Power Distribution/Interconnection Tables

DD11-DK Backplane

Color	Voltage	J1 & J3	J2 & J4	Backplane Slot	Row	Pin
BLACK	GROUND	8,9	8,9	all	a,1	C7
RED	+5 V	1,4,12	1,4	all	all	A2
GRAY,WHITE	+15 V	2	5	all	C	B7
BLUE,GREEN	-15 V	N/C	11,15	all	C	B2

N/C = Not Connected.

DB11-CK Backplane

Color	Voltage	J14	Backplane Slot	Row	Pin
BLACK	GROUND	7,8	all	all	C2
RED	+5 V	1,4,12	all	all	A2
GRAY,WHITE	+15 V	2,5	all	C	B7
BLUE,GREEN	-15 V	11,15	all	C	B2

BA11-KM / BA11-KK Backplane J19 Interconnections

Color	Use	J19	Backplane Slot	Row	Pin
BLACK	GROUND	1	all	a1	c2
BROWN	+5V	2	all	C	D1
VIOLET	DC LO	3	all	C	A1
YELLOW	AC LO	4	all	C	V1

NOTE

The backplane pins listed do not necessarily show where voltage is present for A&B pins. Refer to the appropriate W/L for all connections.

DEVL'-HK Backplane

Color	Voltage	J12	Backplane Slot	Row	Pin
BLACK	GROUND	7,8	all	all	C2
RED	+5 V	1,4,13	all	all	A1
GRAY	+12 V	2	A	01	S1
			B	01	a1
--	-12 V	*	A-B	02	D2

* +12 V is generated from +15 V on the QBus Connector module, M9403, which is installed in slot 1 of the backplane. Refer to drawing CS-M9403-0-1 for wire details. There is a 3 Amp fuse mounted on this module at the + 12 V output.

NOTE

The following adjustment must be made when the H7140 power supply has been replaced. Measure the voltage level at pin AD2 (or BD2), of QBus slot 1 (where the M9403 paddle card is installed). Adjust the trimpot on the M9403 module until the voltage measures +12 vdc ± 0.05 V.

3.10 RL02 (CVT) POWER INFORMATION

The RL02 disk drive and BA11-AL expansion drawer require a Constant Voltage Transformer (CVT) to condition the incoming ac power. The CVT is mounted in the front end cabinet at the top rear of the RL02 (refer to the Power System Component Location diagram). Switched ac power is the input to the CVT from the 876-A power controller and output ac power is provided for the RL02 disk drive and the H7140-CA (in the BA11-AL).

NOTE

The CVT provides 110V output for both 50Hz and 60Hz systems and as such, the input VOLTAGE SELECTOR at the rear of the RL02 disk drive MUST BE SET AT 110V and the VOLTAGE RANGE SELECTOR must be set at NOM in all cases.

The same is true for the H7140-CA power supply in the BA11-AL OMNIBUS Expansion Drawer, which must always have the input voltage selector set at 110V.

CHAPTER 4

INITIALIZATION AND BOOTSTRAP TROUBLESHOOTING PROCEDURES

4.1 INTRODUCTION

This section has two objectives:

1. Describe the major events that occur during the system power-up and boot sequence.

2. Provide a set of troubleshooting procedures that will aid in isolating faults that may occur during the power-up and boot sequence.

The first objective is satisfied with a time line flow chart that illustrates the major visible events that take place during the power-up and bootstrap process, and the approximate time between events. Following the flow chart is a brief explanation of the internal (non-visible) events that take place between the major events, including pointers to the appropriate troubleshooting procedure. The second objective is satisfied by a set of troubleshooting flow charts and procedures.

NOTE

This section is still under development

4.2 BOOT SEQUENCE TIME LINE FLOW DESCRIPTION

1. POWER ON

   a. AC power is routed to the ac input modules (X and Y) which in turn supply:

      - 300vdc to the main power bus (from modules X & Y).
      - 5 Vdc to the I/O backplane (from module X).
      - +12 Vdc to CPU and Clock Module (from module Y).

   If the ac input module fault LED is set on the CPU, see AC Input Unit Troubleshooting Flow.

b. The BNM and Clock Module check the parallel and serial module keying circuits.

- If parallel key loop failure, set the BNM KEY FAULT LED and abort power-up. See Parallel Key Loop Troubleshooting Flow.
- If serial key loop failure, set the appropriate Clock Module LED and the BNM KEY FAULT LED, and check the key override switch (BNM). If the BNM ELECTRICAL KEY OVERRIDE switch has been depressed, then continue. If not, then abort the power-up. See Serial Key Loop Troubleshooting Flow.

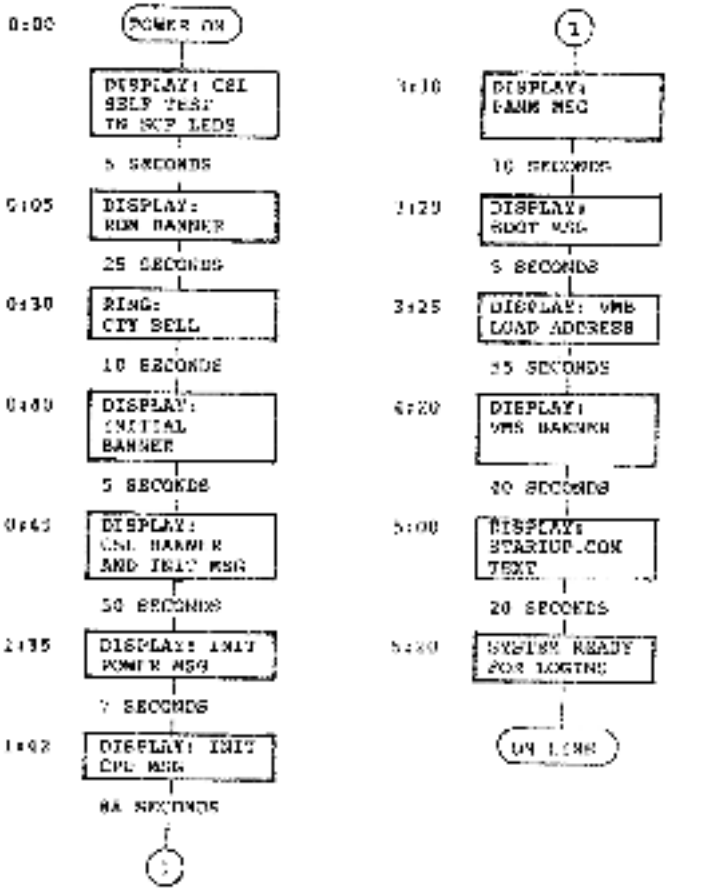


Figure 4-1 Boot Sequence Time Line Flow

c. Turn on Regulator A (+5Vdc to BNK, Console, and I/O backplane).

- If BNK begins self initialization.
- If errors: loop on failing test (console will detect and report).

If Module OK LED is not lit, see Regulator A (+5 Volt Troubleshooting Flow.

d. Console begins self initialization.

- Initialize registers, disable external interrupts, and run Console PROM Self Test (0 through 10).
- If Self Test fails: See Console Initialization and Self Test (1-10) Troubleshooting Flow and STOP the Troubleshooting Chart.

2. DISPLAY ROM BANNER (Console ROM Self Test 11)

a. Execute Self Test (11-35)

- Test 11 will print "VAX 8600" for PROM v36 or earlier, and "PROM Vnn", where nn is the PROM version number, for PROM V37 or higher.
- If Self Test fails: Print a failure message and loop on failing test. See Console Self Test (11-35) Troubleshooting Flow.

3. RUN CTY DIAG.

a. The operator has 5 seconds to interrupt the boot sequence and enter PROM Command Mode by pressing any key on the CTY or FEV.

b. Execute Console PROM SLU2 Tests.

- If Errors: Print error message and abort sequence. See Console QBus Troubleshooting Flow chart.

4. DISPLAY INITIAL BANNER

a. Read and perform checksum of SLU2 Boot Block (Block 0).

b. Initialize console mapping

5. DISPLAY CCF2OLS BANNER and INIT NFS640

a. Load SDS signal name overlays.

b. Start RAP time.

c. If RESUME, restore console and console port parameters.

d. Turn off all SCP LEDs.

e. Initialize system (CPU) Clock

6. DISPLAY INIT ZONES MESSAGE

a. Initialize Console/EMM Communications

b. Read EMM Status

- If no EMM response: Display Message

c. Read EMM Voltage and Temperature parameters

d. Read Regulator E, I and A Status

- If error: Display Message.

e. Read Regulator C, D, E, F, and H Status

- If error: Display Message.

f. Clear all within Parameters

g. Turn on and check regulators: E, F and H, D and E, and C.

- If error: Display Message.

h. Direct EMM to Deassert DC LO

- If error: Display Message.

i. Direct EMM to Deassert DC LO

- If error: Display Message.

j. Direct EMM to clear Magnetic Latches and Air Flow Fault Status.

- If error: Display Message.

k. Direct EMM to enter Default Mode.

7. DISPLAY INIT CPU MESSAGE

- TBD

8. DISPLAY IAPM MESSAGE

- TBD

9. DISPLAY BODY MESSAGE

- TBD

10. DISPLAY VMS LOAD MESSAGE

    - VBO

11. DISPLAY VMS BANNER

    - IDP

12. DISPLAY STARTUP.COM TEXT

    - TBS

13. SYSTEM READY FOR LOGINS

    - TNS

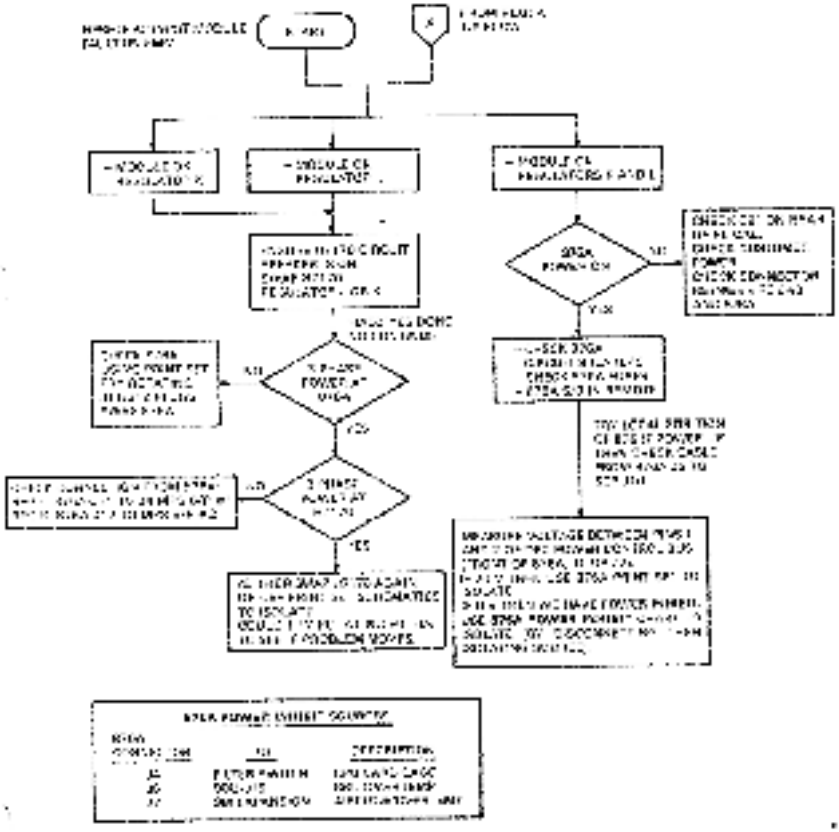


Figure 4-2 AC Input Module (X and Y) Troubleshooting Flow

4.3 MODULE KEYING

4.3.1 Overview

The VAX 8600/8650 system implements an electrical module keying scheme to prevent the unintentional overinsertion of modules when they are inserted in an incorrect slot. The scheme is to detect the incorrect installation of a module prior to system power-on, inhibit system power-on, and flag the user of the condition. The hardware involved in this scheme includes the MPS (EMM), the clock module and the MBH/CPL/ASLH backplanes and modules.

NOTE

Modules within the I/O backplane are not verified under this scheme.

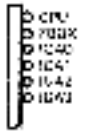
4.3.2 Implementation

Two separate types of module keying loops are used: a technology or parallel loop, and a serial key loop. The parallel loop verifies that modules of incompatible technology are not inserted into the wrong backplanes. The serial key loop verifies the correct slot locations of modules within a particular backplane. The parallel loop must be satisfied before testing the serial loops.

The serial loops are broken down into separate serial key loops, CPU, FBox, and one for each IOA. With the exception of the CPU SERIAL KEY LOOP, the other serial loops must be enabled before they are used (i.e., at least one of the option modules must be inserted in the correct S/P for the serial loop to be enabled).

Refer to the Module Keying Diagram, or, for a more complete description, refer to the EMM Technical Description Manual for a detailed description of key loop operation.

Refer to the Diagram of H7188-A EMM Module-Visual Indicators (Chapter 3) for the location of the KEY FAULT LED on the EMM. The following diagram illustrates the layout of the Serial Key Loop Fault LEDs on the clock module.



CPU Serial Key Loop Fault
FBox Serial Key Loop Fault
IOA0 Serial Key Loop Fault (SBIA 0)
IOA1 Serial Key Loop Fault (SBIA 1)
IOA2 Serial Key Loop Fault (currently N/A)
IOA3 Serial Key Loop Fault (currently N/A)

Figure 4-3 Clock Module Key Fault LEDs

4.3.3 Parallel Key Loop

If the module keying circuits sense a parallel key loop fault, the KEY FAULT LED on the H7189-A (BKP) will be lit and no clock module LEDs will be lit.

NOTE

This condition may also indicate a CPU serial key loop fault if the clock module is not inserted.

The parallel loop will be broken if any of the following conditions exist:

1. A CPU module is inserted in either the ABus or Memory backplanes.
2. A memory module (including the XTM) is inserted in the ABus backplane.
3. An ABus module is inserted in the Memory backplane.
4. The XTM module is missing from slot 9 of the Memory backplane.
5. The XJA module is missing from slot 1 of the ABus backplane.

4.3.4 CPU Serial Key Loop

If the module keying circuits sense a CPU serial key loop fault, the KEY FAULT LED on the H7189-A (BKP) will be lit along with the top LED on the clock module.

The CPU serial loop will be broken if either of the following conditions exist:

1. The clock module is not inserted.
2. Any CPU module is installed in the incorrect slot or a module is not installed. This includes the XJA module in slot 1 of the ABus backplane and either the XTM or FBA module in slot 8 of the CPU backplane. (The XTM and FBA modules are checked by the FBox serial loop).

4.3.5 FBox Serial Key Loop

If the module keying circuits sense an FBox serial key loop fault, the KEY FAULT LED on the H7189-A (BKP) will be lit along with the second LED from the top on the clock module.

The FBox serial loop will be broken if BOTH of the following conditions exist:

1. The FBox serial key loop is enabled – Any CPU module (KS2 ground) is installed in slot 7 or 8 of the CPU backplane.
2. FBN (FJM) and FBA (FTN) are not installed in CPU backplane slots 7 and 8 respectively.

4.3.5 IOA Serial Key Loop

If the module keying circuits sense an IOA serial key cable, the KEY FAULT LED on the E7185-A (ENA) will be lit, along with either the 3rd (IOA0) or 4th (IOA1) LED from the top on the clock module.

The IOA serial loop will be broken at BOIE if the following conditions exist:

1. The IOA serial key loop is enabled - An ABus module is installed in slot 2 or 3 of the ABus backplane.

2. The SDA and SDB are installed in ABus backplane slots 3 and 2 respectively.

NOTE

The above description is for SDIA0. For SDIA1, it would be slots 4 and 5

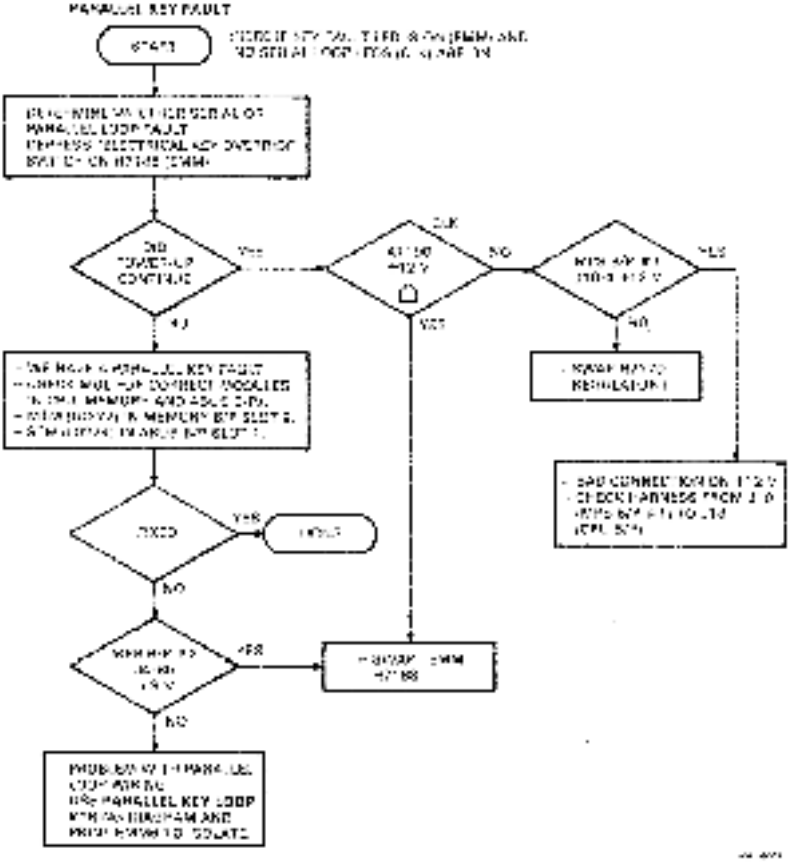


Figure 4-4 Parallel Key Loop Fault Troubleshooting Flow

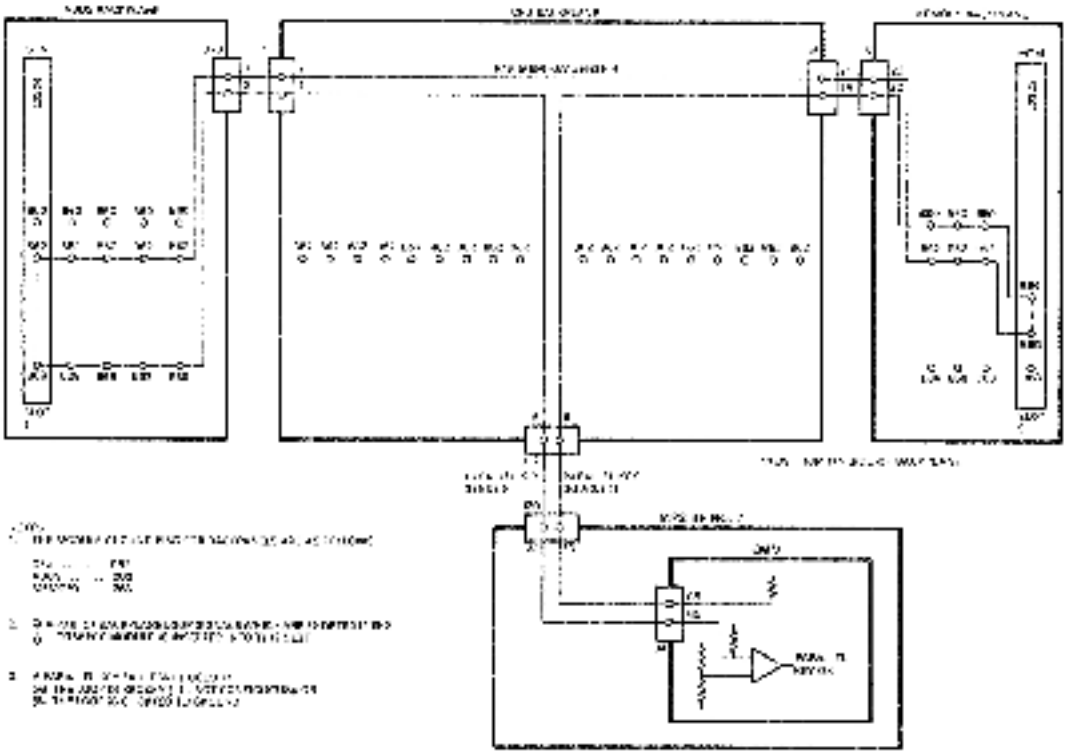


Figure 4-5 Parallel Way Lamp Circuit

INITIAL ISSUE/8001 EXPANDED/CONDITIONED PRODUCE

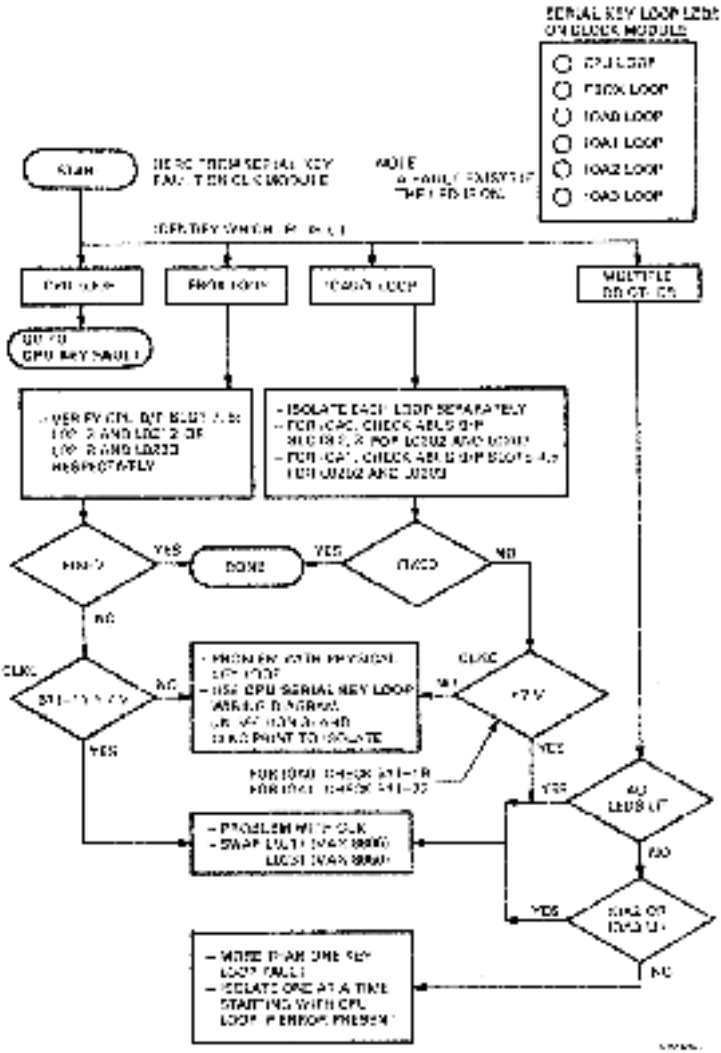


Figure 4-6 Serial Key Loop Fault Troubleshooting Flow

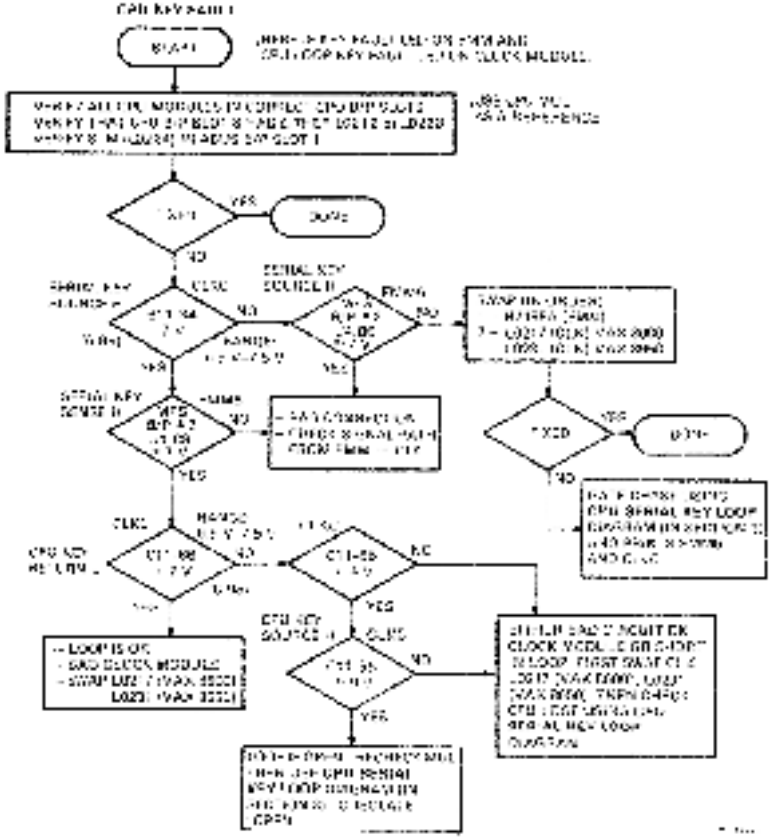


Figure 8-7 CPU Key Fault Troubleshooting Flow

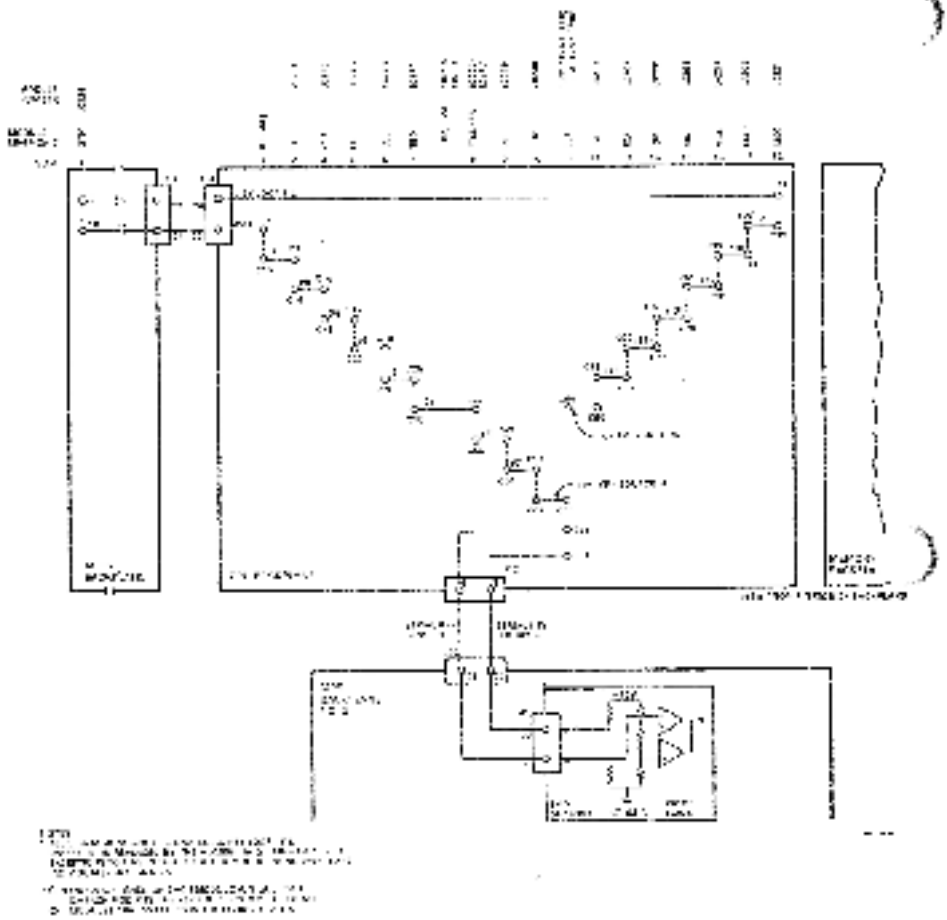
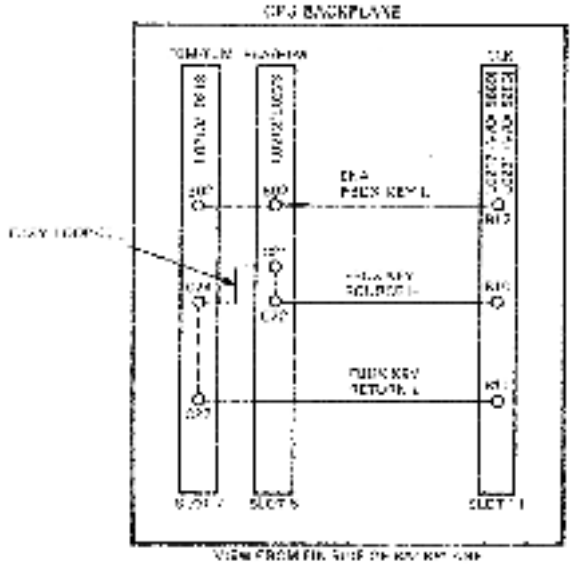


Figure 4-3 CPU Serial Key Load Circuit



NOTES:

1. THE SIGNAL PRA PROM IS INSTALLED WHEN ANY CPU MODULE IS INSTALLED IN SLOT 7 OR SLOT 8.
2. THE KEY LOCK IS ACTIVE IF BOTH THE PRA PROM IS INSTALLED AND THE FRONT PANEL MODULES ARE PRESENT, OR IF NEITHER OF THESE MODULES ARE PRESENT.

Figure 4-9 Mode Serial Key Lock Circuit

Figure 4-13 IBM Serial Key Loop Circuit

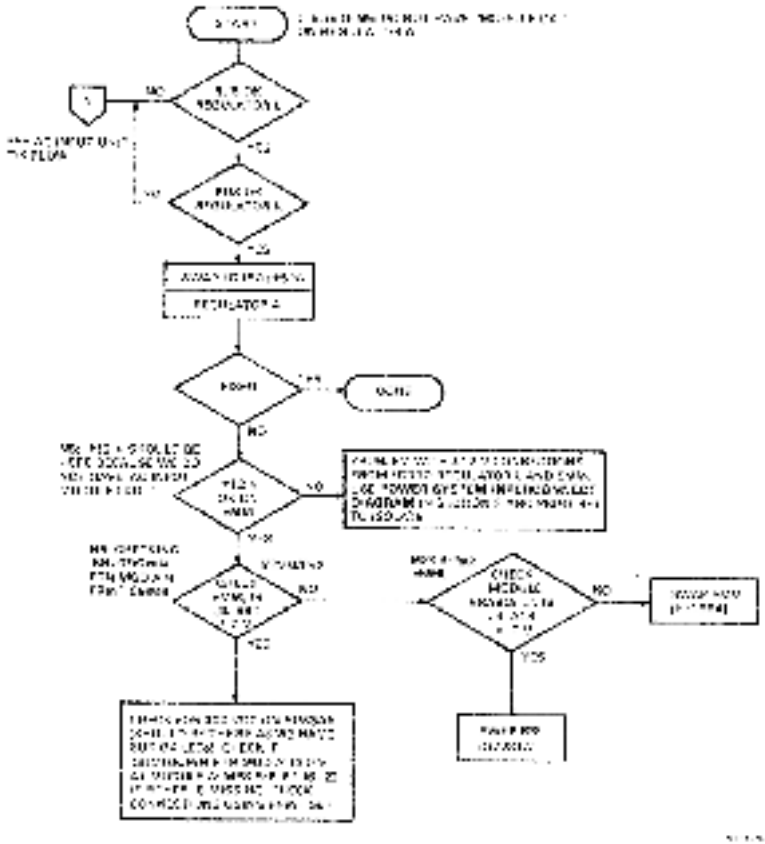


Figure 4-11 Regulator & I+D Voltage Initialization

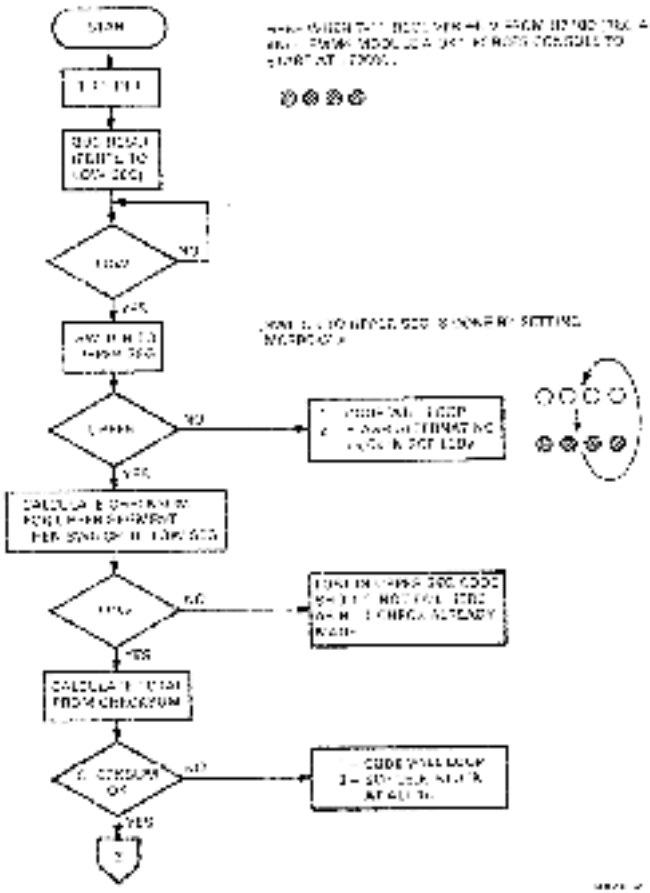


Figure 4-12 Console Initialization and Self Test (1-10) Troubleshooting Flow

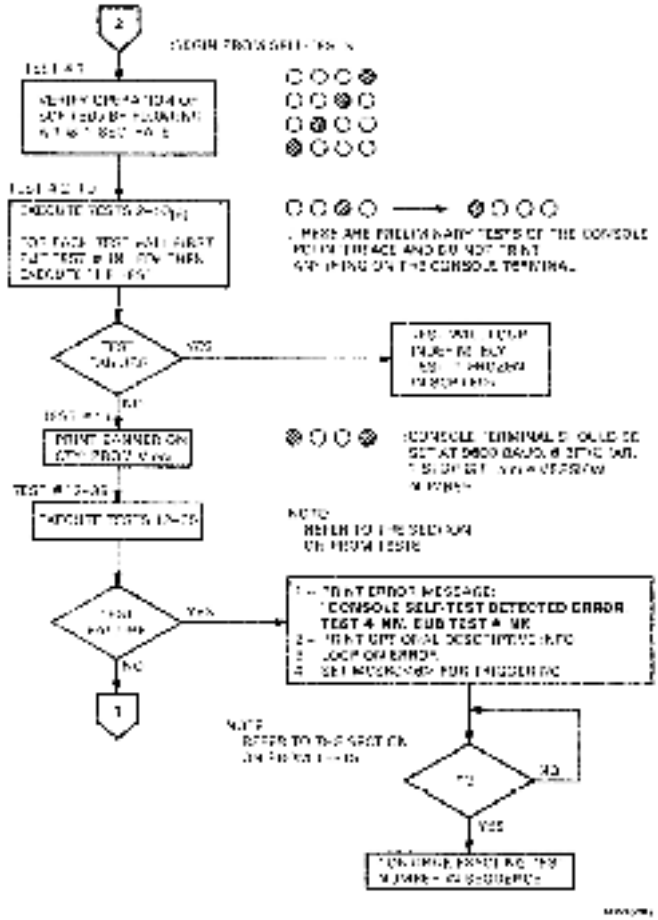


Figure 4-13 Console Self Test (11-55) Troubleshooting Flow

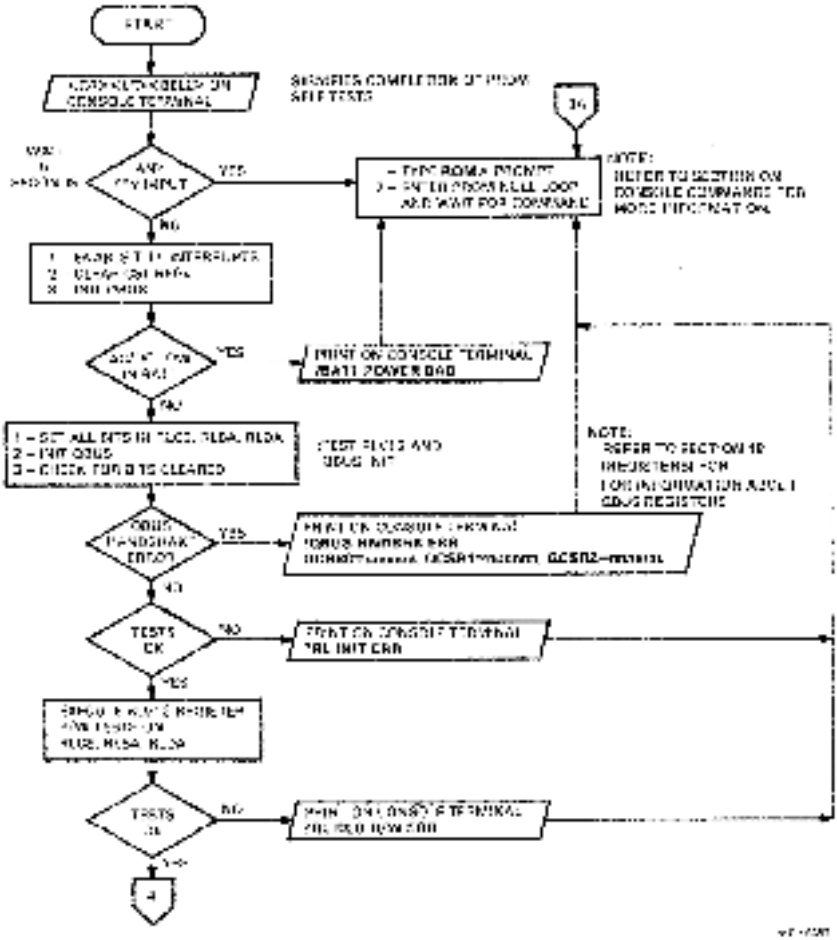


Figure 4-14 Console QBus (BDV2) Troubleshooting Flow (Sheet 1 of 2)

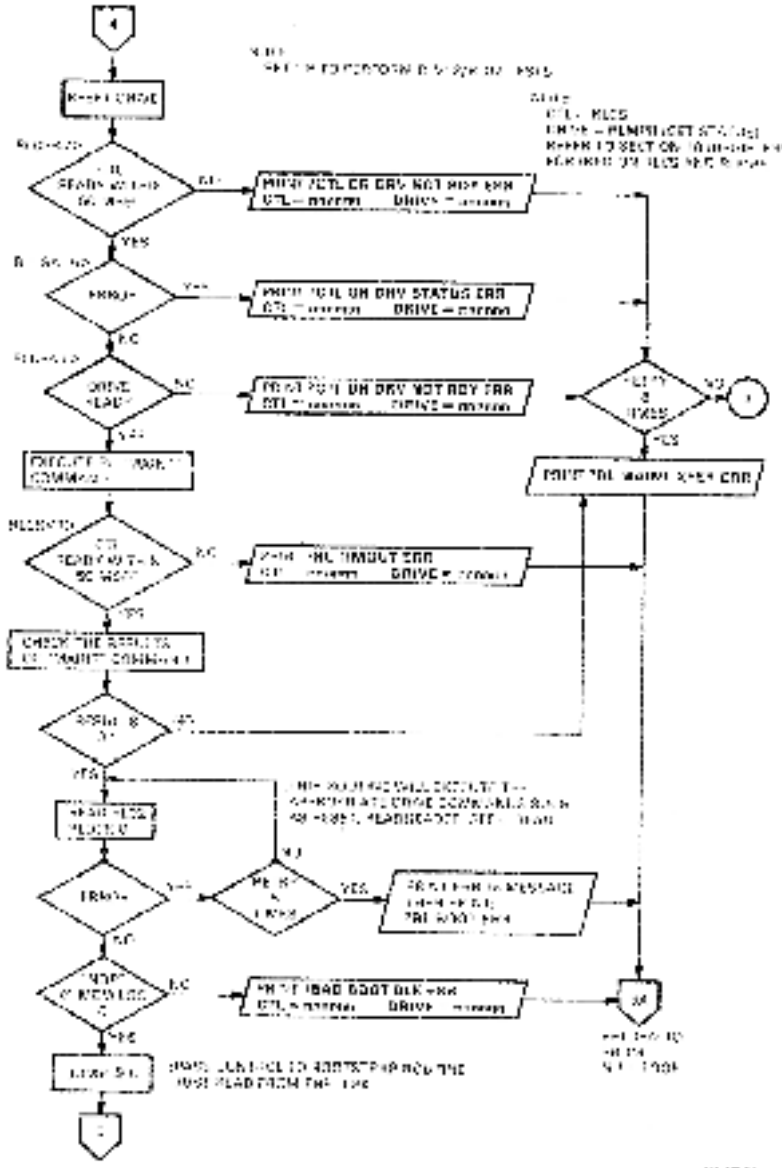


Figure 4-13 Console Diskette (RX01) Troubleshooting Flow (Sheet 2 of 2)

4.4 SCP LED SEQUENCING

The following table outlines the System Control Panel (SCP) LED sequence that transpires during normal system initialization. A section on troubleshooting has been included. During initialization, if the SCP LEDs freeze in a specific state, this chart should assist in isolating the failure. The chart is intended to be used with the Power Up/Initialization flow charts.

NOTE:

In the following chart, a "o" (bullet) indicates that the LED is not lit, and an * (asterisk) indicates that the LED is lit.

SCP (LED) Troubleshooting Chart

SCP LEDs	Action	Troubleshooting
o o o o	Initial condition: No power applied to system.	SCP didn't receive +5 V or console failed to INIT. If you have +5.0 V and EMM MODULE A OK H then swap (7291; check SCP connections or swap SCP.
* * * *	Console initialization: Power has been turned on; regulators 1 & 2 are OK; BMM/CLOCK has verified key loops and regulator 4 has been turned on which provides power to EMM and console. (-11 initializes the console, which lights all SCP LEDs.	The LEDs may all be lit from initial power up, not from CSR initialization. Check BMM MOD OK & H (2236). If it is in +5 V, swap (0201). CSR. If not +5 V, power didn't get to CSR from regulator 4. BMM can pull +5 V down.
* o o * o * o o o * o o * o o o	Console self test # 1: the console verifies operation of the SCP by flashing 1 one through the LEDs. This test should not fail as there are no checks made.	The pattern shifts very fast, too fast to see the pattern. Use a SCOPE POINT/ TEST to verify all LEDs are switched. Problem could be in SCP, (0201), or cable.
o o * o o o * * o * o o o * o * o * * o o * * * * o o o	Console self tests 2 - 8: If any of these self test routines detect an error, the test will loop forever and the LEDs will reflect the number of the test that failed. Because the console terminal has not been tested, this is the only method used to indicate a test failure.	If any of these tests fail they will loop forever. These tests are checking the local SCP. Swap the (0201 module.

* o o *	Console self test (1): This test prints the 'PROM Ver' banner on the CTY (on a current console PROM version). The LEDs will remain in this state until self test completion. This LED state should occur about 3 sec after power-up and will remain for about 22 sec.	The LEDs will remain at 1001 during the execution of the remaining self tests. Test pass/fail criteria: Look at the CTY and verify printout of PROM Ver. If no printout, verify that CTY is set up for 9600 BPS, 8 bits/char, 1 SB, and no parity. Suspect cable to CPU, cable from KEYBC panel to CPU backplane (J10), or LC2C1.
o o o * o o * o o * o o * o o o	Console self test 2-5: SCP light test. It is test will again test 8 one through the SCP LEDs, reading the LEDs state each time.	If any of these tests fail then check the SCP connections; swap the SCP; swap the LU2C1. Should also get an error message.
o o o o	Console self test completed. Self tests have completed successfully.	Even here no errors are reported on the CTY.

o * * o	The PROM code will read the state of the SCP switches and set the LEDs to reflect the state of the Remote port only).
	`----REMOTE Active  (ORBRM)`
	`----REMOTE Enabled (ORBEN)`	

* * * *	After the PROM runs the SLV12/SLU2 tests it will boot the console operating system. At the time you see the 'Initialization' message on the CTY, the SCP LEDs are fully operational and reflect the current state below:
	`----ALARM          (REG)`
	`----REMOTE ACTIVE  (ORBRM)`
	`----REMOTE ENABLED (ORBEN)`
	`--------CPU STATE      (ORBRP)`
		For detailed information of the meaning of these LEDs refer to the KA8E Console Technical Description. Be aware that ALARM may result in a TOTAL-OFF condition.

* * * * o o o o	NOTE: An alternating green & white pattern in the SCP LEDs indicates a PROM Segment Switching error. This error can only occur during PROM code execution.	If, during PROM execution, you get this display, swap the LU2C1 module.

4.5 CONSOLE SELF TESTS

Console Self Tests

Test	Title
0	PROM Checksum Test
1	SCP 300 Channel Test
2	CTY Interface Mode Register 1 Bit Test
3	CTY Interface Mode Register 2 Bit Test
4	CTY Interface Command Register Bit Test
5	CTY Interface Reset Test
6	CTY Interface DataBit Bit Test
7	CTY Interface RxRDY Bit Test
10	CTY 9600 BPS Loop Back Test
11	PROM Version Number Display Test
12	Parity Error Latch Test
13	Parity Circuit Test, Part 1
14	Parity Circuit Test, Part 2
15	56 KB RAM Data/Address Test, Bottom-up
16	56 KB RAM Data/Address Test, Top-down
17	Mapping RAM Location 0 CV Test
20	Mapping RAM Data Test, Bottom-up
21	Mapping RAM Data Test, Top-down
22	Mapping RAM Addressing Test
23	TOY Chip Access Test
24	KCY Interface Mode Register 1 Bit Test
25	KCY Interface Mode Register 2 Bit Test
26	KCY Interface Command Register Bit Test
27	KCY Interface Reset Test
30	KCY Interface TxRDY Bit Test
31	KCY Interface RxRDY Bit Test
32	KCY 1200 BPS Loop Back Test
33	SCP 300 Logic Test
34	CLIS TSTAT Interrupt Test
35	Unexpected Interrupt Test

4.6 VMB UNEXPECTED INTERRUPT TROUBLESHOOTING PROCEDURES

The following procedure may be used to aid in the analysis of boot failures resulting in VMB issuing an UNEXPECTED EXCEPTION message with a HALT PC of 39E.

The procedure patches VMB such that the SCB is initialized to contain VECTOR ADDRESS + 3 for all vectors except those otherwise initialized by VMB (i.e., Machine Check). Normally these vectors would point to the VMB UNEXPECTED EXCEPTION handler.

Execute this procedure as follows:

>>>BOOT/NOSTART	Boot without starting - note load address = 200
>>>@VMBPLT	Execute this procedure to patch and start VMB

If an UNEXPECTED EXCEPTION condition occurs the CPU will halt. The HALT PC will reflect the instruction which caused the exception. ESSW Scratch Pad location 11 will contain the contents of the exception vector. Examine this location and mask out bits <1:0> to determine the vector address.

For example:

```
CPU stopped, SCB vector(3)=1, INVALID ICSM code 07
PC 00302F0C
>>>D/H/F 12
        9 12 00000019
```

The example implies that the instruction which caused the exception is at 302F0C and that we received through SCB% 1C (3*4 with bits <1:0> = 0) indicating a reserved addressing mode.

NOTE

This procedure is only valid for VMB V-4.C2 when it is loaded at 200. If FIND/MEMORY should return any value other than 200 in the SP this procedure can be executed manually by adding the value of (SP - 200) to those addresses used below (2600 and 260A).

Also, if using a different version of VMB, examine VMB.MAP to locate PSECT VBTMEM, assuming that the first three (3) instructions relative to the start of this PSECT do not change. This procedure could be modified to use the PSECT VBTMEM (starting address) + 200 for the base address of 2600 and PSECT VBTMEM starting address +20A for the base address of 260A.

VMB Unexpected Interrupt Patch Procedure

Command	Comments
EX/WORD 2600	The contents of this location should be xxxx03FF reflecting MOVAB BOOTFAULT+1,R5
DEPOSIT * 203A03C	Replace with MOVBWL 203,R6
EXAMINE/WORD 260A	The contents of this location should be 5600 reflecting MOVL R5,-(SP)
DEPOSIT/WORD * 7603	Replace with MOVAL -(R6),-(SP)
SET SNAP OFF	Ensure that the CONSOLE does not create a snapshot file
START 200	Start VMB

4.7 VMB Q/R USAGE

VMB CPR Usage

R0R0 UNUSED

R1 BOOT DEVICE TYPE:

<07:00> BOOT DEVICE TYPE CODE:

0	MASSBUS device (RM02/3, RP04/5/6/7, RM80)
1	RK06/7
2	RL01/2
3	UDA (almost an RA80) on 11/730
7	TU45
32	SDI or CI
64	Console block storage device

<15:08> reserved for future expansion

<31:16> device class dependent (R1@VMB_R1.STD)

UNIBUS - optional vector address; 0 implies use of the default vector

MASSBUS = not used

R2 BOOT DEVICE'S BUS ADDRESS:

11/780 & 11/730

<31:04> MBA
<03:00> TR number of adapter

11/750 -

<31:24> MBZ
<23:00> address of the I/O page for the boot device's adapter

VAX 8600 -

<31:06> MBZ
<05:04> A-BUS ADAPTER NUMBER
<03:00> TR number of the adapter

R3 BOOT DEVICE UNIT OR CONTROLLER NUMBER OR PORT NUMBER:

- UNIBUS: <31:18> MBZ
  <17:00> UNIBUS address of the device's CSR
- MASSBUS: <31:04> MBZ
  <03:00> Adapter's controller/formatter number
- CI: <31:08> MBZ
  <07:00> HSC port number (station address)

R3 BOOT DEVICE UNIT NUMBER:

R4 LOGICAL BLOCK NUMBER:

- logical block number to boot from if bit 3 is set in R5 (not supported on 11/730)

R5 Bootmgr Boot Control Flags:

Bit	Title and Description
0	Conversational boot. At various points in the system boot procedure, the bootstrap code solicits parameters and other input from the console terminal. If the DIAG is also set, then the diagnostic supervisor should enter MSBU mode and prompt user for devices to test.
1	Debug. If this flag is set, VMS maps the code for the XDELTA debugger into the system page tables of the running system.
2	Initial breakpoint. If SDBSV_DEBUG is set, VMS executes a BPT instruction immediately after enabling mapping.
3	Secondary boot from boot block. Secondary bootstrap is a single 512-byte block, whose LBN is specified in R4.
4	Diagnostic boot. Secondary bootstrap is image called [SYSMAINT]DIAGBOOT.EXE.
5	Bootstrap breakpoint. Stops the primary and secondary bootstraps with a breakpoint instruction before testing memory.
6	Image header. Takes the transfer address of the secondary bootstrap image from that file's image header. If SDBSV_HDRXFR is not set, transfers control to the first byte of the secondary boot file.
7	Memory test inhibit. Sets a bit in the PFN bit map for each page of memory present. Does not test the memory.
8	File name. VMB prompts for the name of a secondary bootstrap file.
9	Halt before transfer. Executes a HALT instruction before transferring control to the secondary bootstrap.
10	No BPB detection. (Not implemented) Intended to tell VMB not to read a file from the boot device that identifies bad or reserved memory pages, so that VMB does not mark these pages as used in the PFN bitmap.)
11	Multi-port memory only. Specifies that multi-port memory is to be used for the total Exec memory requirement. No local memory is to be used. This is for tightly-coupled multi-processing.
12	Multi-port memory combined. Specifies that multi-port memory should be used in addition to local memory, as though both were one single pool of pages.

15 Execute extensive memory test.
Specifies that a more extensive algorithm be used when testing main memory for hardware uncorrectable (HRD) errors.

14 Requests use of NA780 memory if MS780 is insufficient for booting. Used for 11/782 installations.

15 Used by Diagnostic Supervisor.

16 Specifies that memory pages with correctable (CRD) error not be discarded at bootstrap time. By default, pages with CRD errors are removed from use during the bootstrap memory test.

<31:28> Directory Number.

Specifies the top level directory number for system disks with multiple systems.

The hardware or the CONSOLE program sets up the next 3 registers after a system crash or power failure:

R10 - halt PC
R11 - halt PSL
AP - halt code

SP - <bascaddress - A200> of 64 Kb of good memory

CHAPTER 5

ERRORS AND ERROR ANALYSIS

5.1 SYSTEM FAULT ISOLATION

The information which follows is extracted from the KA86 System Fault Isolation Manual (EK-V86VB-MM). Refer to this manual for a detailed description, if required.

The following figure shows the overall organization of error detection, error handling, and error reporting for the KA86 processor. As shown in the figure, each box (XBox, IBox, EBox, and FBox) has its own error detection network. These detection networks constantly monitor for error conditions and report them directly to the EBox (except for the SSIA). The SSIA monitors for external and internal detected errors. External detected errors are reported to the Cbox Interrupt Arbitration Logic, and internal detected errors are reported to the MBox, which sends them to the EBox.

5.1.1 EBox Interrupt and Exception Arbitration Logic

All errors, except for MBox and CBox control store parity errors, are reported to the EBox Interrupt and Exception Arbitration Logic. This logic prioritizes the errors and generates a starting address which is the starting address of a special EBox microcode routine designed to handle error conditions.

5.1.2 Error Handling Microcode (EHM)

The EHM reads the state of major CPU Control and Status Registers and places the result in the EBox Scratch Pad RAM. This data is referred to as the Machine Check Stack Frame. The Interrupt/Exception microroutine that showed pushes the Machine Check Stack Frame onto the interrupt stack and calls the VMS Machine Check Handler (MCHK).

5.1.3 VMS Machine Check Handler

The MCHK pops the Machine Check Stack Frame off the interrupt stack and puts it in the System Event Buffer, queues the buffer to be appended to the System Event File (SYSERR.SYS), and either Megaswerks or Bugs depending on the severity of the error.

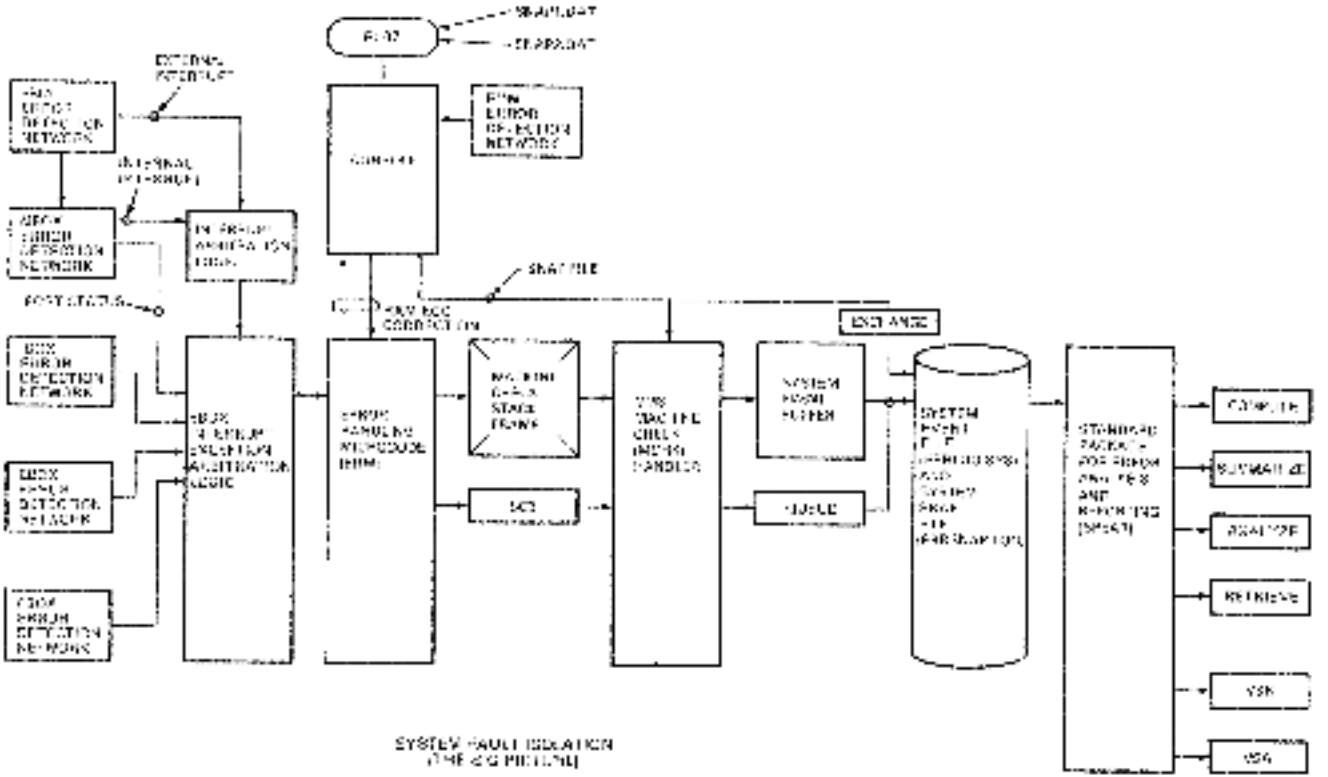


Figure 5-1 System Fault Isolation - Functional Block Diagram

5.1.4 SPEAR (Standard Package for Error Analysis and Reporting)

SPEAR is a maintenance tool designed to sort, analyze, and display the contents of the System Event files.

5.1.5 Keep Alive Fail (KAF)

The CPU has two KAF mechanisms; one is used by the console to monitor the operation of the CPU, and the other is used by VMS to monitor operation of the console.

5.2 CONSOLE SNAPSHOT FILE INFORMATION

Upon detection of a Keep Alive failure, the Console will read and log the status of the machine into a snapshot file on the console RL02. Upon VMS system reboot, the snapshot file is transferred from the console to the system error log on the system disk. Since the current system Error Reporting Facility (ERF) does not recognize the format of this snapshot error file, there are a number of additional utility programs that must be used to interpret and analyze the snapshot file. The following text describes how they operate and the flow chart which follows illustrates the utilities.

NOTE

If you need to decode/translate a SNAPSHOT file that is on the RL02 disk, you can:

1. Reboot the system, which will transfer the valid SNAPSHOTs to the system disk. VER may be used to translate the SNAPSHOT.

2. Call the EDC and ask them to uplink dump and translate the SNAPSHOT.

3. Print the SNAPSHOT file using the Console command "BOOT SNAPn.DAT" where n is 1 or 2. Use the System Fault Isolation manual, Appendix B, to decode the console printout.

5.2.1 Guide to Decoding/Analysis of a SNAPSHOT

There is a command procedure that may be run which automatically performs many of the functions shown on the chart. A simplified method using this command procedure follows. Note that this procedure assumes that you have already set up the necessary programs and defaults to run VSBELD.

1. After VMS is rebooted, the snapshot is transferred to ERRSNAP.LOG;nn

2. Rename the ERRSNAP.LOG;nn file to a unique name such as FILENAME.LOG

3. From the FIELD account, enter the command:

   $ @SYS$UPDATE sys$error:FILENAME.LOG

4. Either:

   a. Read and decode the output files FILENAME.HOR, FILENAME.BIG and FILENAME.REG; or

   b. Contact the EAC and request assistance (for SNAPSHOT file analysis)

5.2.2 Reference Documentation

VAX 8600 Console Software Specification, Revision 0.1
VSS Package Documentation
KA86 System Fault Isolation Manual, EK-0860S-MM

5.2.3 SNAPSHOT File Flowchart Notes

The following notes support the 'Console SNAPSHOT File Information Flow Chart'. The note number in brackets [n] refers to the circled numbers on the chart.

1. Console RL02 snapshot files: SNAP1.DAT and SNAP2.DAT. These are the original snapshot files as taken by the console hardware upon the detection of a Keep Alive Failure. These files reside on the console RL02. Refer to KA86 System Fault Isolation Manual for a description of the format of these files.

2. Transferring the snapshot file from the console RL02 to a remote system. If the remote system has a logging capability (e.g., VMS with an accodial DF03 using the 'SET HOST/DTE3' command), it can connect to the 8600 via the remote port and by using the 'SBOM SNAPn.DAT/ASCII' command, type the snapshot file. This will in effect transfer the snapshot, in ASCII format, to the remote system. This method is also useful for accessing snapshots when a system is down solid.

3. Filenm.TXT is the output file created when the ASCII snapshot file is spline loaded using the 'SBOM SNAPn.DAT/ASCII' command. This file will not have carriage control (it will contain embedded <CR>s,<LF>s).

   Note that when SNAPTO.EXE is run to convert this ASCII file into a binary file, an interim step takes place where the .TXI file is converted to a .DAT file using VSROUT.PDL. This step is necessary to create a file with carriage control.

4. SNAPTO.EXE is a program which converts the ASCII snapshot file (created via the 'SBOM SNAPn.DAT/ASCII' command) into a binary file which is suitable for input to the VSROUT and VSA programs.

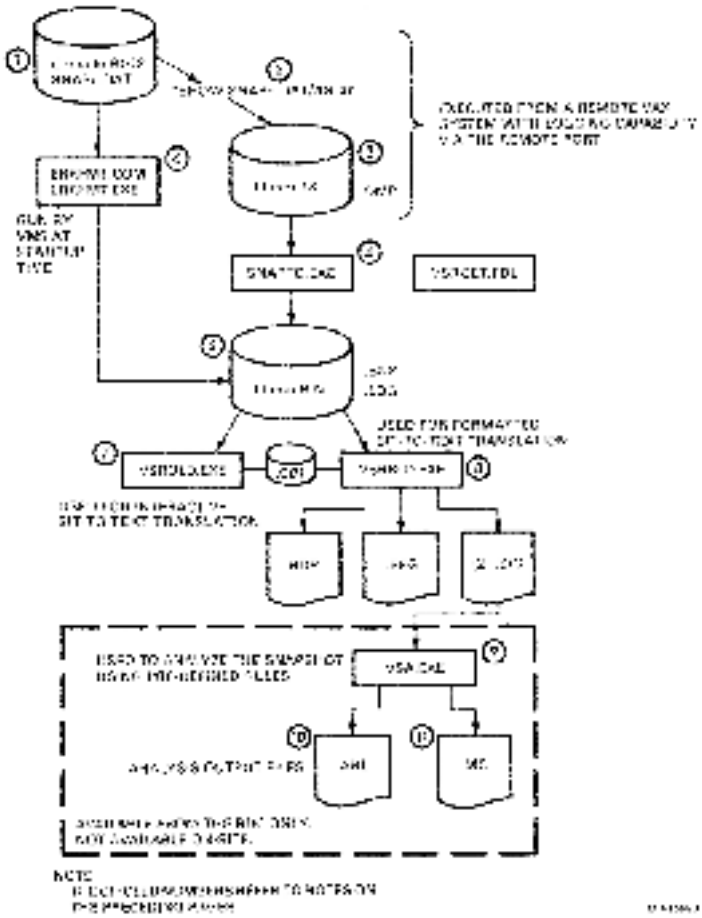


Figure 5-2 Console Snapshot File Information Flow Chart

5. Filename.bin, or ERRSNAP.LOG/ERR. These are the binary snapshot files. These files may be the output from the SNAPSHOT.EXE program or they may be part of a system error log. In the latter case, the binary snapshot files are actually not part of the system error log, rather each snapshot file is a separate file (ERRSNAP.LOG/ERR) with linkages to the system error log file.

6. At VMS system startup time, ERRPKT.COM is executed and determines whether or not valid snapshot files exist on the console RL02 pack. If a valid file is present, the file is transferred to the SYS$ERROR area (filename: ERRSNAP.SNP;n), an entry with a pointer to the snapshot file is made in the system error log file (this is done by ERLVMT.EXE), and the console RL02 snapshot file will be invalidated.

7. VSBOLD.EXE is the original bit-to-text translation package that was used to translate these snapshot files. This program is menu driven and can be used for selective translation of the snapshot files. Note that the translate options include only signal names and block dumps; there are no register decodes or sorting done by VSBOLD.EXE. The input file for VSBOLD.EXE is the binary snapshot files and the output may be either TT: (driven or file oriented (refer to the VSBOLD menu for more details). The ASCII file signal name tables (*.CDE) are also required as input to VSBOLD.EXE.

8. VSPBLD.EXE is used to provide formatted ASCII output from a binary snapshot file. Using a binary snapshot file as input, and the .CDE files as data, VSPBLD generates three output files:

   - A "filenam.HDR" file containing information on each record within the snapshot file, including the SEALED record.
   - Two "filenam.SIG" files which contain a list of all of the visibility points recorded in the snapshot file. The second version of this file will have the signals sorted alphabetically.
   - A "filenam.DMP" file containing formatted bit-to-text translation of all records (except the module visibility channels) within the snapshot file.

   The combination of all three of these output files represents all of the information available in the snapshot file. There is no error analysis done during the creation of these files, rather it is up to the Field Engineer to manually decode and analyze the snapshot or to contact the Remote Diagnosis Center for assistance.

9. VSA.EXE is used to ANALYZE snapshot files. The input to VSA.EXE includes the ASCII signal file generated by VSBOLD.EXE (*.SIG) as well as the original binary snapshot file (ERRSNAP.SNP;n). VSA.EXE will generate two ASCII output files, one containing stall analysis (if applicable) and signals of importance, the second file containing the analysis of embedded stack frames within the snapshot file.

NOTE

VSA.EXE is Company Confidential and the program is NOT AVAILABLE ON SITE. To run VSA, it is necessary to contact the Remote Diagnosis Center, which will uplink load the SNAPSHOT file then run VSA.EXE at the remote site.

10. "filenm.ANL", the output of the SNAPSHOT analysis program, will contain the following information:

    - A list of applicable SCRIL theories (if any)
    - SCRIL theory descriptions (optional)
    - Signals of importance (includes error signals that were found in the true state)
    - Default signal list (a list of nice-to-know signals such as UPCs etc.).

11. "filenm.MEM", the second output of the SNAPSHOT analysis program, includes the output of a machine check stack frame if one was found embedded within the snapshot. For example, if the snapshot was the result of a Double Error Halt and a stack frame was found in the EBox scratch pad registers, VSA.EXE will extract and analyze the stack frame and then output the results to this file.

5.1 CONTROL STORE PARITY ERROR CORRECTION

Control store parity errors are prioritized by the XBox, and transmitted as a three-bit code to the console. The presence of any of these three bits will generate a console interrupt, the code indicating which control store is in error. The console will disable external interrupts and suspend the KA? clock to prevent the XBox from interrupting the console and to prevent a keep alive fail condition while the EBox is not executing macro instructions. The console will correct the control store parity error, if possible, and set up the information for the XBox CSES register; the RAM ID, syndrome, CS or DRAM address, and the uncorrectable bit, if correction fails.

If the control store parity error occurs in the EBox, the clock to the control store data register is disabled preventing the XBox from executing any more micro-instructions.

After the console corrects the control store parity error and sets up the CSES register, it will reset the control store parity error and generate an XBox microtrap (CSPE RST LST CYC) with the CSB XDE control channel and "CI XMHANG RESET H". The microtrap will call the XMem error handling microcode.

If the control store parity error is in the IBox or FBox, the EBox will have been informed by the error handling microcode. The EBox will transfer the XMSR to XMCR, which will provide the XMachine to generate the three-bit code for the console. The EBox microcode will then clear XBUSY (07) (XHN-) and loop while waiting for TXNE to set.

After the console has corrected the parity error and set up the CSES, it will clear the XHNI bit, which will allow the EBox error handling microcode to proceed with error logging.

If the control store parity error is in the XBox, there will be no attempt to restart the system because the XBox has already executed the faulty microinstruction.

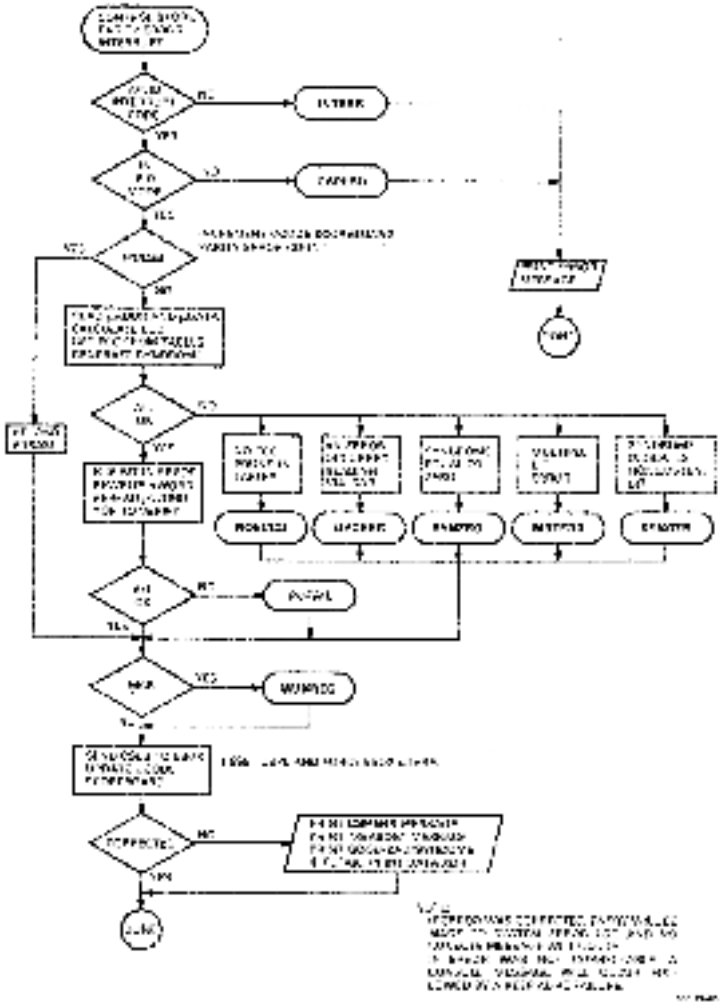


Figure 5-3 Control Store Parity Error Simplified Flow Chart

5.4 SYSTEMATIC INFORMATION GATHERING PROCEDURES

The following methods can be used to gather information concerning the hardware status after an error.

You should first try to use Method 1. If unable to use this procedure due to something being hung, try Methods 2, 3, 4, and 5. When doing specialized initialized, each succeeding method destroys more and more information.

5.4.1 Method 1

NOTE

GMAP must have been set to "OFF" prior to the failure.

```
>>> ^P

>>>> STOP CPU

>>>> HIC or IMIC        Find the hung microcode loop

MMSM>>>> SPACE

SMSM>>>> SPACE

SMMP>>>> SPACE

SMSM>>>> SPACE (etc.)

SMSM>>>> CR             Or get out of space bar step mode

>>>> UNJAM              Restores CSM without disturbing anything

>>>> .SET QUIET OFF

>>>> @STAPIN           Executes the command file that dumps the FSC
                       locations which may contain a Machine Check

>>>> E MAPEN           Is memory management turned on?

>>>> E SP
```

Use the appropriate command, depending upon the contents of "MAPEN".

```
If MAPEN = 1
>>>> E/V/XMS:100 @     Dump the STACK

If MAPEN = 0
>>>> E/P/XEXT:100 @    Dump the STACK
```

Continue dumping other registers. The following examples are system configuration dependent.

>>>> @KDSBIO	Execute the command file to dump SBIA IO registers
>>>> @KDSBI1	Execute the command file to dump SBIA #1 registers if the system has two SBIs
>>>> E/X xxxxxxxx	Execute the appropriate console commands to examine the SBI KDM70 and peripherals

5.4.2 Method 2

>>> DSB

>>>> RESET	This will reset the CPUs, so the TB is lost
>>>> UNJAM	Restart CPU without disturbing anything
>>>> SET QUIET OFF	
>>>> @STKDMP	Dumps ALL locations which may contain a Machine Check
>>>> @KDSBIO	Execute the command file to dump SBIA #0 registers
>>>> @KDSBI1	Execute the command file to dump SBIA #1 registers if the system has two SBIs
>>>> E/X xxxxxxxx	Execute the appropriate console commands to examine the SBI KDM70 and peripherals

5.4.3 Method 3

>>>>INIT/CPU	Resets the CPU. Only the I/O registers will be valid
>>>> @KDSBIO	Execute the command file to dump SBIA IO registers
>>>> @KDSBI1	Execute the command file to dump SBIA #1 registers if the system has two SBIs
>>>> E/X xxxxxxxx	Execute the appropriate console commands to examine the SBI KDM70 and peripherals

5.4.4 Method 4

>>>>INIT/CPU	Resets the CPU. Only the I/O registers will be valid
>>>>INIT/PASM	SBIA registers are no longer valid
>>>> E/X xxxxxxxx	Execute the appropriate console commands to examine the SBI KDM70 and peripherals

5.4.3 Method 3

>>>INIT/CPU	Resets the CPU. Only the I/O registers will be valid
>>>INIT/PARM	UBIA registers are no longer valid
>>>> BOOT	The UNIBUS and MASSBUS peripheral devices may contain valid information, but everything else is lost
>>>> E/X xxxxxxxx	Execute the appropriate console commands to examine the CSR addresses and peripherals

5.5 EXAMPLE

The following command file should be put on each VAX 8600/8650 system RL02 pack.

The purpose of the file is to provide a quick method of executing a hardware dump for keep-alive failures, and whenever you have chosen not to use the "BOOT and VMB/VSA" facilities.

NOTE

> The portion of this command file that examines the I/O controllers, and peripheral devices, is system configuration dependent. Therefore, it will have to be modified to reflect the configuration of the system it is to be used on.

After this file is dumped onto the RL02 pack (using the EXCHANGE copy command) you can use it by typing in "@DUMP" to the console prompt. For this information to be valid, "BOOT" must be set "OFF" prior to the failure.

This file must be typed in exactly as is, except for the comments, which may be left out. If you do not wish to type all this in, the command file is available on the SMSI. You can then dump it on tape for transporting to the different sites. To have a copy mailed to your node, contact your local support organization.

```
!         Hardware Register & Stack dump command file
DEBUG
SET OUTPUT OFF
SET ABORT OFF
SET BASE 8
!
!  Get microcode loop PCs
!
STOP CPU
!
WIDE
!
WIDE
!
WIDE
!
WIDE
```

```
MICRO

MICRO

MICRO

MICRO

MICRO

MICRO

MICRO

MICRO

UBHALT
HALT
!
!  Examine SCRATCH PADS - data is as indicated only if
!  a machine check was detected and the CPU was halted
!  prior to re-entering MACROCODE program.
!
E/ESC 17        ! Byte count in Stack Frame
E/ESC 18        ! ERR.STS  -  Error Handling Microcode Status
E/ESC 19        ! VMQSAV   -  sometimes holds VMA for EBox port
E/ESC 1A        ! EBCS     =  EBox Control and Status
E/ESC 1B        ! EDPSR    -  EBox Data Path Status
E/ESC 1C        ! CSLINT   -  Console Interrupt Word
E/ESC 1D        ! IBESR    -  IBox Error Summary
E/ESC 1E        ! MDAWD1   -  Last (or 2nd to last) word sent to
                !             MBox (write)
E/ESC 1F        ! MDAWD2   -  Last (or 3rd to last) word sent to
                !             MBox (write)
E/ESC 20        ! IVASAV   -  IBox Virtual Address
E/ESC 21        ! VIBASAV  -  VA of next IBUF Fork request to fill
                !             IBuffer
E/ESC 22        ! MSASAV   -  MBox Execution/Result storage PC
E/ESC 23        ! IBASAV   -  PC of instruction Op Port is working on
E/ESC 24        ! CPC      -  PC of instruction be evaluated in PBUFF
E/ESC 25        ! MBTAT1   -  MBox Status
E/ESC 26        ! MBTAT2   -  MBox Status
E/ESC 27        ! MDECC    -  MBox ECC status
E/ESC 28        ! MERG     -  Memory error status
E/ESC 29        ! CSHCTL   -  Cache control
E/ESC 2A        ! MEAR     =  Address in PA latch when error occurred
E/ESC 2B        ! MEDR     -  Data word in use when error occurred
E/ESC 2C        ! SHXERR   -  EBox Xreqs (will be FFFFFFFF if no errors)
E/ESC 2D        ! CSES     -  CS correction (will be FFFFFFFF if no
                !             errors)
E/ESC 2E        ! PC       -  Program PC when error occurred
E/ESC 2F        ! PSL      -  Processor Status Longword
!
!  Examine all Scratch Pads, CPU registers, and I/O registers.
!  Also verify contents of all Control Store RAMs.
!
E/ESC/N:FF 0    ! Dump all EBox Scratch locations
!
P/G/N:F 0       ! General Purpose Registers
!
E/I/N:24 0      ! Internal Registers
!
```

```
E CPC           ! IBox current PC
K CSHCTL        ! MBox Cache Control
E CSCS          ! Control Store Error Status Summary
E CSCINT        ! Console Interrupt Status
E EBCS          ! EBox Control and Status
E EDPSR         ! EBox Data Path Status
K EMD           ! IBox EMB
E EVASAV        ! IBox EVA.SAV
E IBSTR         ! IBox Error Status
K ISASAV        ! ISA.SAV in IBox
E IVASAV        ! IVA.SAV in IBox
E MBDR          ! MBox Error Data
K MEAR          ! MBox Error Address
E MSTAT1        ! MBox Status Register #1
E MSTAT2        ! MBox Status Register #2
K VIRASAV       ! IBox VIBA.SAV
E VPCSAV        ! Valid PC bits
!
!  Examine the STACK
E SP
K/V/HEX;100 @
!
!  Examine all ABox adapter, SSL Memos, and peripheral registers
!
K 20080000      !SBIA #0 Configuration
E 20080004      !SBIA #0 Control and Status
K 20080008      !SBIA #0 Error Summary
E 2008000C      !SBIA #0 Diagnostic Control
K 20080010      !SBIA #0 BNAI Command/Address
E 20080014      !SBIA #0 BNAI ID
K 20080018      !SBIA #0 BNAA Command/Address
E 2008001C      !SBIA #0 BNAA ID
K 20080020      !SBIA #0 BNAB Command/Address
E 20080024      !SBIA #0 BNAB ID
K 20080028      !SBIA #0 BNAC Command/Address
E 2008002C      !SBIA #0 BNAC ID
K 20080030      !SBIA #0 SILO
K 20080034      !SBIA #0 SILO
E 20080038      !SBIA #0 SILO
K 2008003C      !SBIA #0 SILO
E 20080040      !SBIA #0 SILO
K 20080044      !SBIA #0 SILO
E 20080048      !SBIA #0 SILO
K 2008004C      !SBIA #0 SILO
E 20080050      !SBIA #0 SILO
K 20080054      !SBIA #0 SILO
E 20080058      !SBIA #0 SILO
K 2008005C      !SBIA #0 SILO
E 20080060      !SBIA #0 SILO
K 20080064      !SBIA #0 Error
E 20080068      !SBIA #0 Timeout Address
K 2008006C      !SBIA #0 Fault/Status
E 20080040      !SBIA #0 SILO Command
K 20080044      !SBIA #0 Maintenance
E 20080048      !SBIA #0 UNIBUS
K 2008004C      !SBIA #0 Quadclear
]
```

```
C 22080000      !CSR'A #1 Configuration
E 22080004      !SBIA #1 Control and Status
K 22080008      !MSCR #1 Error Summary
E 2208000C      !SBIA #1 Diagnostic Control
C 22080010      !SBIA #1 DMAI Command/Address
E 22080014      !SBIA #1 DMAI ID
K 22080018      !SBIA #1 DMAA Command/Address
E 2208001C      !SBIA #1 DMAA ID
K 22080020      !SBIA #1 DMAP Command/Address
E 22080024      !SBIA #1 DMAB ID
K 22080028      !SBIA #1 DMAC Command/Address
E 2208002C      !SBIA #1 DMAC ID
E 22080030      !SBIA #1 SILO
K 22080030      !SBIA #1 SILO
E 22080030      !SBIA #1 SILO
K 22080030      !SBIA #1 SILO
E 22080030      !SBIA #1 SILO
K 22080030      !SBIA #1 SILO
E 22080030      !SBIA #1 SILO
K 22080030      !SBIA #1 SILO
X 22080030      !SBIA #1 SILO
S 22080030      !SBIA #1 SILO
K 22080030      !SBIA #1 SILO
K 22080032      !SBIA #1 SILO
E 22080030      !SBIA #1 SILO
K 22080030      !SBIA #1 SILO
S 22080030      !SBIA #1 SILO
S 22080034      !SBIA #1 Scrap
K 22080038      !SBIA #1 Timeout Address
E 2208003C      !SBIA #1 Fault/Status
K 22080040      !SBIA #1 SILO Compare
S 22080044      !SBIA #1 Maintenance
K 2208004A      !SBIA #1 CNIMR
E 2208004C      !SBIA #1 Quadrilear
!
E 20006000      ! UBA Configuration
E 20006004      ! UBA Control
K 20006008      ! UBA Status
E 2000600C      ! UBA Diagnostic Control
E 20006010      ! UBA Failed MAP Entry
E 20006014      ! UBA Failed UNIBUS Address
K/N-7 20006020  ! UBA Buffer Selection Verification Registers 0-7
E/N:3 20006030  ! UBA BR Receive Vector Registers 4-7
K/N=F 20006040  ! UBA Data Path Registers 0-15
!
S 2001C000      ! CI Configuration
E 2001C004      ! CI Control and Status
!
E/PARM/K:FF 0
SHOW POWER
SHOW PARITY
SHOW UCODE
SHOW VERSION
SHOW CLOCK
!
!  This next step will take a few minutes.
!
VERIFY          ! Verify the contents of all Control Stores
!
SET QUIET OF
```

5.6 PROGRAM TO SCAN MEMORY FOR PARITY ERRORS (SCAN.COM)

```
 >>> UNJAM

>>>> SET SNAP OFF

>>>> DEP R0 0                ; setup starting address

>>>> DEP R1 0                ; temporary storage

>>>> DEP 100 175180FF        ; MOVL (R0)+,R1

>>>> DEP 104 CBB1F0FC        ; JUMP @#100

>>>> DEP 108 0               ; HALT

>>>> DEP BCBB 200            ; place SCB starting at location 200

>>>> DEP 204 207             ; Setup Machine Check trap catcher.

>>>> DEP 254 257             ; setup SBE trap catcher.

>>>> DEP DBL 41C000C         ; allows interrupts from "10" up.
                             ; MSCP interrupt is at "15".

>>>> DEP SP 1000             ; setup Stack Pointer.

>>>> DEP ESSR 0              ; Clears all CPU error bits.

>>>> ST 100
```

```
; Should halt with ESC #12, and ESC #C4, containing a 207.
; This indicates that a machine check has occurred. Verify
; that the Machine Check was the result of a MXM by dumping
; ESC and checking errors.


; If ESC #12 contains 254 and ESC #C4 contain a 257, the
; problem was a single-bit error (SBE).

>>>> E/ESC 12

>>>> @STEPFM

; or dump all ESC locations with the following command:

>>>> E/ESC/K:FF 0
```

5.7 VMS AND UBA SYSTEM EVENT FILE (ERRLOG.SYS)

The VMS Operating System maintains a System Event File called ERRLOG.SYS. The file is located in the SYS$SYSROOT:[SYSERR] directory and is used to record errors, status changes, messages, and other events that occur during system operation.

Each time one of these events occur, the normal operation of VMS is interrupted and a special routine is called to handle the event. The routine requests a System Event Buffer and then gathers pre-defined information about the event (e.g., system status, hardware and software registers, etc.) and puts it in the buffer. Once the buffer is built the routine queues a request to append the buffer to the System Event File. When the queue is processed the buffer is appended to SYS$SYSROOT:[SYSERR]ERRFILE.SYS.

Two programs (ANALYZE/ERRORLOG and RETRIEVE - a Spear library function) are available to translate the contents of the System Event File into ASCII reports. Both of these programs use the Event Record Formatter (ERF) to translate the entries in the Event File. Therefore, regardless of which program you use the format of the translated entries will be the same. The main difference between the two programs is the command syntax, the selection criteria, and the format of the summary reports they produce. In addition to translating system event file entries Spear is capable of analyzing the contents of the event file and calculating system availability.

5.7.1 ANALYZE/ERRORLOG

ANALYZE/ERRORLOG uses a non-interactive command syntax. That is, the Command, Qualifiers and Arguments, are entered in a single string. The Qualifiers allow you to select specific entries from a binary System Event File and either produce a separate binary event file that contains only those entries, or translate the entries and produce an ASCII Report. For a complete description of this utility, including more information about the ANALYZE/ERRORLOG command and its qualifiers , see the VAX/VMS Utilities Reference Volume.

5.7.2 SPEAR Library Functions

SPEAR is a maintenance tool specifically designed to help Field Service Engineers sort and analyze the contents of System Event Files. There are two versions of Spear: SPEAR Basic and SPEAR Extended.

5.7.2.1 SPEAR Basic - SPEAR Basic is available at all sites that have a DEC Maintenance Contract. This version of SPEAR consists of five programs:

- Instruct - Instruct is a computer based instructional program that is designed to help new users learn to use the SPEAR Library Programs. In addition to explaining the SPEAR Programs, Instruct also describes the organization of system event files and includes a review of some of the most common troubleshooting approaches.

- Compute - Compute is designed to use the contents of the System Event File to calculate system availability and effectiveness. The Compute report can be used (in part) to determine if the system is approaching the point where corrective maintenance will soon be required.

- Summarize - Summarize is designed to summarize the contents of the system event file. The Summarize report can be used to determine whether the CPU or one or the I/O subsystems needs further investigation.

- Retrieve - Retrieve is a bit-to-text translator. It is designed to extract specific entries from the System Event File and produce either a brief or full translation of the event. This information can be used to investigate the causes of CPU and I/O failures.

- VSR (Venus Snap File Report Builder) - This program was added to the Basic SPEAR Library specifically to support VAX 8600/8650 Systems. Like Retrieve, VSR is a bit-to-text translator. VSR, however, is designed to translate SNAP files. A SNAP file is a file built by the console as a result of a Keep Alive Fail condition.

5.7.2.2 SPEAR Extended - SPEAR Extended is only available at Remote Diagnosis Centers. In addition to the programs that are available in SPEAR Basic, Spear Extended includes:

- Analyze - Analyze is designed to analyze the contents of large System Event Files and identify the most probable cause of certain failures. Analyze evaluates the events in a System Event File against a set or if-then isolation theories. If the Events support a theory then the theory is displayed for consideration by the Engineer at the RD center.

- VSA (Venus Snap File Analysis Builder) - VSA is similar to Analyze except it is designed to analyze the contents of System SNAP Files.

CHAPTER 6

DIAGNOSTICS

NOTE

Chapter 6 will cover:

- KMM SHOW TEST
- VIA (PROM) SELF TEST
- PROM COMMAND SET
- CONSOLE MODULE DIAGNOSTIC (EDDBA)
- MICRO-HARD-CORE DIAGNOSTIC (EDKAA)
- MBC COMMAND SET
- MICRO DIAGNOSTICS
- MICRO DIAGNOSTIC COMMAND SET
- MICRO DIAGNOSTIC COMMAND FILES
- DIAGNOSTIC SUPERVISOR (EDSAA)
- MACRO DIAGNOSTICS

The remaining command sets, GENERAL, MACRO, and MSX are located in Chapter 10, Console Commands.

6.1 EMM Self Test

When the VAX 8xxx is first turned on and power is applied to the EMM, the EMM will automatically run a set of Self Tests. The tests are listed in Table 6-1. The EMM uses the Error Codes listed in Table 6-2 to inform the console of error conditions.

6.1.1 Prerequisites

None

6.1.2 EMM Self Test Descriptions

Table 6-1 EMM Self Test

TEST	DESCRIPTION
ROMCHK	ROM Checksum Verification - Compute the ROM checksum and verify that it is correct.
UTEST1	Data test for MODE registers - Tests that MODE1, MODE2, and COMMAND registers can hold 125 and 252 This test doesn't verify that they are separately addressable. See Note 1.
UTEST2	Test Addressability of USART Registers - Verifies that MODE and COMMAND registers are independently addressable and that they can hold data without interfering. See Note 1.
UTEST3	Test USART in Local Loopback - Tests status register and data registers at 9600 baud (We don't test for some errors, e.g., framing and overrun, which could be forced, since the protocol will exclude incorrect messages because they don't checksum.) See Note 1.
RTEST1	Parity Circuit, Part 1 - The following algorithm is used to test parity for each RAM.

1. Disable RST 0 ON PARITY ERROR. If a RST 0 occurs because of a parity error it indicates that the RST 0 ON PARITY ERROR circuit could not be disabled.

2. Select parity to be generated (GEN PAR)

3. Write DATA pattern to first RAM location (20000)

GEN PAR	TEST DATA	PAR RAM
ODD	10001010	YES
ODD	01010101	YES
EVEN	11101100	NO
EVEN	00110011	NO

4. Read pattern back from first RAM location

5. Test returned DATA pattern. If error, reports:

   - BAD DATA RAM LOCATION

6. Test for expected PAR ERR condition. If error, reports:

   - BAD PARITY GENERATOR CIRCUIT, or
   - BAD PARITY RAM LOCATION, or
   - BAD PARITY-CHECK CIRCUIT

7. Repeat above steps for all TEST DATA

PTEST2 Parity Circuit, Part 2 - The following algorithm is used to test parity for each RAM.

1. Enable RST 0 ON PARITY ERROR
2. Set indicator so "RST 0" routine will know that this is a forced error.
3. Select ODD parity generation
4. Write something into FIRST RAM location and read it back
5. If no RST 0 occurs, report:

   - "PARITY ERROR FAILED TO RESET CPU"

RTEST3 RAM Data Test - The following algorithm is used to test data for each RAM.

1. Disable RST 0 ON PARITY ERROR
2. Select EVEN parity generation
3. Write data pattern into RAM location:

TEST DATA	DAA BIT	PAR ERR
00110011	0	NO
01010101	0	NO
10101000	1	NO

4. Read RAM location and compare result.

   - If compare fails in low-order nibble (bits 0-3) reports: "BAD DATA RAM LOCATION, LOW-ORDER NIBBLE"
   - If compare fails in high-order nibble (bits 4-7) reports: "BAD DATA RAM LOCATION, HIGH-ORDER NIBBLE"

5. Test PAR ERR flag. If PAR ERR reports "BAD PARITY RAM LOCATION"

6. Repeat steps for all patterns and all RAM locations

RTEST1 RAM Addressing and Cell Interference Test - The following algorithm is used to test data for each RAM.

1. Disable RPL 0 ON PARITY ERROR circuit.

2. Generate RAM data pattern from column 'a'.

   - For '-1' values, select ODD parity generation.
   - For '0' values, select EVEN parity generation.

TEST DATA

RAM location	a	b	c	d	e	f	g	h	i	j
0000	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
	0	-1	-1	-1	-1	-1	-1	-1	-1	-1
	-1	0	-1	-1	-1	-1	-1	-1	-1	-1
	0	0	-1	-1	-1	-1	-1	-1	-1	-1
	-1	-1	0	-1	-1	-1	-1	-1	-1	-1
	0	-1	0	-1	-1	-1	-1	-1	-1	-1
	-1	0	0	-1	-1	-1	-1	-1	-1	-1
	0	0	0	-1	-1	-1	-1	-1	-1	-1
	-1	-1	-1	0	-1	-1	-1	-1	-1	-1
	0	-1	-1	0	-1	-1	-1	-1	-1	-1
	-1	0	-1	0	-1	-1	-1	-1	-1	-1
	0	0	-1	0	-1	-1	-1	-1	-1	-1
	-1	-1	0	0	-1	-1	-1	-1	-1	-1
	0	-1	0	0	-1	-1	-1	-1	-1	-1
	-1	0	0	0	-1	-1	-1	-1	-1	-1
	0	0	0	0	-1	-1	-1	-1	-1	-1
	-1	-1	-1	-1	0	-1	-1	-1	-1	-1
	0	-1	-1	-1	0	-1	-1	-1	-1	-1
	-1	0	-1	-1	0	-1	-1	-1	-1	-1
	0	0	-1	-1	0	-1	-1	-1	-1	-1
	-1	-1	0	-1	0	-1	-1	-1	-1	-1
	0	-1	0	-1	0	-1	-1	-1	-1	-1
	-1	0	0	-1	0	-1	-1	-1	-1	-1
	0	0	0	-1	0	-1	-1	-1	-1	-1
	-1	-1	-1	0	0	-1	-1	-1	-1	-1
	0	-1	-1	0	0	-1	-1	-1	-1	-1
	-1	0	-1	0	0	-1	-1	-1	-1	-1
	0	0	-1	0	0	-1	-1	-1	-1	-1
	-1	-1	0	0	0	-1	-1	-1	-1	-1
	0	-1	0	0	0	-1	-1	-1	-1	-1
	-1	0	0	0	0	-1	-1	-1	-1	-1
0031	0	0	0	0	0	-1	-1	-1	-1	-1
	etc.									
	-1	-1	0	0	0	0	0	0	0	0
	0	-1	0	0	0	0	0	0	0	0
	-1	0	0	0	0	0	0	0	0	0
1FF9	0	0	0	0	0	0	0	0	0	0

3. Verify RAM data pattern. If error, reports:

   - "BAD DATA RAM ADDRESS SELECT"

4. Check for expected PAR ERR condition: 1-1 data should have PAR ERR, 00 data should not have PAR ERR. If failure, report:

   - "BAD PARITY RAM ADDRESS <address>"

5. Repeat steps for each column

T57T8T Test 5.6 Interrupt - Tests that if the output of the LAT DC LO flip flop is asserted, then it generates a 5.6 interrupt request to the CPU and that no interrupt will occur. This is not meant to be a comprehensive test, nor is a fault considered to be fatal.

IT5V8: Test 7.6 Interrupt - Tests that the RX CLK output of the USART and the 4 bit CTR work to cause interrupts at 860 microsec intervals. Failures are either that no interrupt request could be seen at the CPU when it should have existed, that when it did exist at the CPU no interrupt occurred, or that it did not time correctly.

NOTE

1. UBRR (Error Handler for all USART Errors) - If a USART test fails, toggle the Magnetic Disk Display once then loop on failing test. In order to toggle the disks, LAT AC LO must be deasserted. In a power up situation, this will not hurt the main CPU because the registers have not been turned on yet. If the error is intermittent, the console may have to reload the AC LO and then initiate it again after LAT DC LO has been deasserted.

2. The self test checks the USART, ROM, and RAM. Any errors detected during the self-test are reported by DIVDR, which sends A unique character for each error (see Table G-2 below), waits 5 seconds, and returns to the failing test (loop on Error).

6.1.3 KMM Error Code Descriptions

Table 6-2 KMM Error Code Descriptions

Code	Error Code Description
9	Module A not OK, went at end of self-test
A	Completed self test successfully
B	Parity ERROR, either due to Parity Generator or Parity Checker or BAD Parity RAM
C	RAM Data ERROR - LOW order nibble
D	RAM Data ERROR - BOTH nibbles (might be addressing problem)
E	RAM Data ERROR - HIGH order nibble
F	Parity Error did not cause RESET to PC = 0
G	Parity RAM Data ERROR
H	Bad Parity ERROR expiration before accessing RAM (in SETCODE)
I	ROM had Checksum ERROR
J	LAT PC 5.5 did not set 5.5 interrupt request. No loop on error
K	5.5 Interrupt request did not cause interrupt. No loop on error
L	7.5 Interrupt set after CPU reset
M	1/2 CHAR timer did not set interrupt request (or not at the wrong rate)
N	7.5 Interrupt request did not cause interrupt
O	7.5 Interrupt did not reset

6.2 Z-11 (PROM) SELF TEST

During power-up, the Z-11 microprocessor executes the PROM self-tests to validate the console module hardcore logic. If the Self Test runs successfully then the PROM Code will sound the Bell on the CTY (RTY). If the operator responds within 5 seconds by pressing a any key on the keyboard the PROM Code will display the PROM Prompt; BDK> and enter Command Mode. The PROM Command set is described in Table 6-3. The individual tests are described in Table 6-4.

6.2.1 Prequisites

Successful KMM Self Test and Power-up.

6.2.2 PROM Command Set

The PROM code signals the completion of the self-tests by sending the bell character to the CTY. At that point the operator can type any key to signal the PROM to enter the command loop. The PROM code indicates its readiness for input by displaying the BDK> prompt.

While in the null loop, the PROM code may service the CTY or will as the RTY (provided the front panel switch is in the REMOTE position and a remote connection has been made). The command parser in the PROM is extremely simple. It accepts single-character commands and alphanumeric arguments. The available commands are listed in Table 6-3.

Table 6-3 PROM Command Set

Command	Description
B	B<cr> Boot the console software from the RL02. This command begins by testing the interface between the console and the RL02 controller. The sequence of tests follows. 1. Checks for AC LOW and DC LOW conditions in the BA11. 2. Checks that console-generated BUS INIT resumes the RL02 controller and clears all RL02 controller registers. 3. Checks that RLCS, RLBA, and RLDA will retain a full complement of patterns. 4. Performs an RL02 MAINTENANCE TRANSFER to verify DMA logic between the console and RL02, and other logic on the RL02 controller. 5. Checks that the first word read in from the RL02 boot block contains "240" (NOP). If any of the above checks fail, a message is reported to both the CRT and RT1, and the B command aborts. There is no test isolation capability. If a drive or controller status error is detected during the actual booting of the device, a number of retries are made.
D	D addr data<cr> Deposit the data word to the specified address. The address must be an even number.
E	E addr<cr> Examine the data word at the specified address. Odd addresses are made even by dropping the low order bit.
S	S addr<cr> Start the T-11 execution at the specified address. The address must be an even number. The command-argument delimiter is optional.
T	T file.ext<cr> Without any argument, this command returns the PROM self-tests and then loads and runs the console diagnostic program (ZKDDA) for two passes. If a filename is specified, the self-tests are not run and the file is loaded and started at location 200. The default filename extension is .SAV.

NOTE

The console diagnostic ZKDDA invoked with the PROM T command tests the RT1 interface which causes the remote port to be disconnected. This prevents running ZKDDA from the remote port.

O	O addr data<cr> This command allows direct deposits to QBus registers. The addresses 174400, 174402, 174404, and 174406 are the BLCS, BLBA, BLBC, and BLMPR registers, respectively. All other addresses are selected.
R	R addr<cr> This command allows direct examines of QBus registers. The addresses 174400, 174402, 174404, and 174406 are the BLCS, BLBA, BLBC, and BLMPR registers, respectively. All other addresses are selected.
V	V<cr> With this command, the PROM code enters a loop which allows characters to pass directly between the CRT and RTY. The escape sequence "<CTRL/P><CTRL/X><CTRL/P>" can be entered from either input device to force the PROM code to exit this mode.
X	X<cr> This command supplies (less any switches) with the description of same in Chapter 11 of Digital Standard 012. It is intended to be used for binary data transfers between T-11 memory and a computer using the remote port input. It is not intended for human use and will work properly only when issued from the RTY dev ce.

6.2.3 T-11 (PROM) Self Test Descriptions

NOTE

Failures in tests 1 through 10 are indicated in the System Control Panel LEDs.

When test 11 and, test failures are reported on the CTY (TTY) because the CTY interface, the cable, and the printer are assumed to be operational.

Table 6-4 T-11 (PROM) Self Test Descriptions

TEST	DESCRIPTION
00	PROM CHECKSUM TEST - An additive checksum of both the lower and upper segments of the PROM is calculated. The result should be 177777. On error, a "BE ," is executed. LED state = 1111
01	SCP-BUS CHANNEL TEST - A floating 1 pattern is spitted through the LEDs at a fairly slow speed so the operator can tell if there is a LED failure. The test checks most of the BUS control logic and the SCP bus logic which is used by subsequent tests to report test numbers. The test is scan-coded and cannot fail. LED state = 0001 --> 0010 --> 0100 --> 1000

02 CTY INTERFACE MODE REGISTER 1 TEST (TEST) - Alternating 01010101 and 10101010 patterns are written and read from the CTY's PCI_MODE_REGISTER_1.

LED state = 0010

03 CTY TRANSLATE MODE REGISTER 2 TEST - Alternating 01010101 and 10101010 patterns are written and read from the CTY's PCI_MODE_REGISTER 2.

LED state = 0011

04 CTY INTERFACE COMMAND REGISTER BIT TEST - Both a 01010101 and a 10101010 pattern are written and read from the CTY's PCI_COMMAND_REGISTER.

LED state = 0100

05 CTY INTERFACE RESET TEST - A pattern is loaded into the PCI_COMMAND_REGISTER and a PCI RESET is performed. If the command register clears, then the PCI's RESET input is connected and working.

LED state = 0101

06 CTY INTERFACE TXRDY BIT TEST - The local PCI is configured to run in its normal operating mode (9600 baud, 8-bit, no parity, 1 stop bit). The TXRDY bit in the PCI status register is then expected to set within 100 ms. If ok, the PCI transmit register is loaded and the TXRDY bit is expected to clear immediately.

LED state = 0110

07 CTY INTERFACE RXRDY BIT TEST - The local PCI is reinitialized, but this time in loop, /wrap back mode. A character is transmitted and the RXRDY bit in the PCI status register is expected to set. When the RECEIVED_CHARACTER_REGISTER is read, the RXRDY bit is expected to clear.

LED state = 0111

10 CTY INTERFACE LOOPBACK TEST - LED state = 1000 (this is essentially the same as the previous test except that several loopback transmissions are performed to ensure the PCI and crystal can handle it.

LED state = 1001

11 CTY BAUDEX TEST - LED state = 1001 This is an unmonitored test that relies on the operator to observe the system behavior.

LED state = 1111

12 PARITY ERROR LATCH TEST - Tests whether the latch responsible for indicating parity errors can be set and cleared directly.

13 PARITY CIRCUIT TEST, PART 1 - Simultaneously tests that RAM location 0 will hold a 01010101 pattern. If all right, parity RAM location 0 is expected to contain 1.

14 PARITY CIRCUIT TEST, PART 2 - Simultaneously tests whether RAM location 0 will hold a 10101010 pattern. If all right, parity RAM location 0 is expected to contain 0 (since the "force parity error" bit in MCSR6 was set prior to depositing the pattern).

15 56 KB RAM DATA/ADDRESS TEST, BOTTOM-UP - A modified moving-inversions test is performed on the first 56 Kbytes of physical RAM. The test verifies all data faults that are likely to occur in the console RAM configuration, as well as verifying all addressing faults in the positive (incrementing) direction. On error, the address, expected data, and received data are displayed.

16 56 KB RAM DATA/ADDRESS TEST, TOP-DOWN - A modified moving-inversions test is performed on the first 56 Kbytes of physical RAM. The test verifies all data stuck-at faults likely to occur in the console RAM configuration, as well as all addressing faults in the negative (decrementing) direction. On error, the address, expected data, and received data are displayed.

17 MAP RAM LOCATION 0 CK TEST - A loop is first performed that uses the first mapping RAM location (MAPR00) to initialize all of physical memory with 0's and good parity, and places each 4 Kbyte page number in the first location of its own 4 Kbyte page boundary. The process is repeated to check that the first byte of each page contains the correct value.

20 MAPPING RAM DATA TEST, BOTTOM-UP - This test uses the memory pattern verified by the previous test to check that all mapping RAM locations can access pages uniquely, in the positive direction.

21 MAPPING RAM DATA TEST, TOP-DOWN - This test uses the memory pattern verified by the previous test to check that all mapping RAM locations can access pages uniquely, in the negative direction.

22 MAPPING RAM ADDRESSING TEST - This test uses the memory pattern verified by the previous tests to check that all mapping RAM locations are themselves uniquely addressable. The memory mapper is then turned off.

23 IOY CHIP ACCESS TEST - This test checks that registers in the communication IOY chip can be accessed to verify that the IOY chip is receiving power from the +5 V signal input from the BUS.

24 RTY INTERFACE MODE REGISTER 1-BIT TEST - Normal operation patterns are loaded into the RTY's PCI_MODE_REGISTER_1 and checked.

25 RTY INTERFACE MODE REGISTER 2-BIT TEST - Normal operation patterns are loaded into the RTY's PCI_MODE_REGISTER_2 and checked.

26 RTY INTERFACE COMMAND REGISTER BIT TEST - Both a 01010101 and a 10101010 pattern are written and read from the RTY's PCI_COMMAND_REGISTER.

27 RTY INTERFACE RESET TEST - A pattern is loaded into the PCI_COMMAND_REGISTER and a PCI RESET is performed. If the command register clears, it means the RTY's RESET input is connected and working.

30 RCV INTERFACE TXRDY BIT TEST - The remote PCI is configured to run in its normal operating mode (1200 baud, 8-bit, no parity, 1 stop bit). The TXRDY bit in the PCI status register is then expected to set within 100 ms. If ok, the PCI transmit register is loaded and the TXRDY bit is expected to clear immediately.

31 RCV INTERFACE RXRDY BIT TEST - The remote PCI is reinitialized, but this time in local loopback mode. A character is transmitted and the RXRDY bit in the PCI status register is expected to set. When the RECEIVED_CHARACTER_REGISTER is read, the RXRDY bit is expected to clear.

32 RCV INTERFACE LOOPBACK TEST - This test duplicates the function of the previous test in order to verify that the remote PCI can handle consecutive character transmissions.

33 SCP SDA LOGIC TEST - This test checks the continuity of the SDA control channel on the SCP. This test will affect the state of the front panel LEDs, but is done so quickly it is unnoticeable.

34 "CLR INTRPT" INTERRUPT TEST - This test checks that the CLR INTRPT interrupt input to the console is not asserted. This interrupt is not testable through the PSM, so it must be cleared before the console program will run.

35 UNEXPECTED INTERRUPT TEST - This test checks that after a full console reset there are no pending (or stuck) interrupts to the T-11 microprocessor. This ensures the proper start-up of ST and the console kernel. <CTRL/Z> - Exit failed test loop is also available.

NOTE

Allowable control characters for tests 12 - 35 are:

- <CTRL/S> - Stop test execution.
- <CTRL/Q> - Resume test execution.
- <CTRL/C> - Suppress Error Report

6.1 CONSOLE MODULE DIAGNOSTIC (EDDEA)

EDDEA is a T-11 based console module diagnostic designed to test all the logic on the console module, including the console functions of the clock, CPU, CBus, Q-Bus, and local and remote serial interface logic. EDDEA does not test any logic in the VAX CPU logic itself.

6.1.1 Prerequisites

The T-11 (PROM) Self Test must have run error free.

6.1.2 EDDEA Switch Register Options

BIT	SWITCH	DESCRIPTION
<15>	100000	Exit subtest loop and proceed
<14>	040000	Loop on failing subtest
<13>	020000	Inhibit baud display
<12>	010000	Enable trace
<11>	004000	Inhibit test iterations
<09>	001000	Loop on test number in <07:00>
<07:00>	000xxx	Select test xxx for loop

6.1.3 EDDEA Control Characters

Control Key	Function
<CTRL/C>	Restart EDDEA
<CTRL/S>	Suspend output
<CTRL/Q>	Resume output
<CTRL/U>	Erase current line
<CTRL/G>	Enter SWR-CHANGE mode
<CTRL/P>	Exit to the PROM CIT

6.1.4 EDDEA Test Descriptions

TEST NO.	DESCRIPTION	NO. OF SUBTESTS
DDB Control Logic Tests		
001	SILO Register Test (CL20)	3
002	BDOR Register Test (CL20)	3
003	RCVS Register Test (CL20)	3
004	CL18 Baud Clock Test (CL18)	2
005	DR Single Step Mode Test (CL19)	3
006	XMR Normal Mode Test (CL19)	2
007	DDB Loopback Test, Normal Mode (CL19)	9

Qbus 808 Channel Continuity Tests

```
010*  SIB Channel 00 Continuity Test (CL21)                    3
011*  808 Channel 01 Continuity Test (CL21)                    2
012*  808 Channel 02 Continuity Test (CL21)                    2
013*  808 Channel 03 Continuity Test (CL21)                    1
014*  808 Channel 04 Continuity Test (CL21)                    2
015*  808 Channel 05 Continuity Test (CL21)                    1
016*  808 Channel 06 Continuity Test (CL21)                    2
017*  808 Channel 07 Continuity Test (CL21)                    1
020*  808 Channel 08 Continuity Test (CL21)                    1
021*  808 Channel 09 Continuity Test (CL21)                    1
022*  808 Channel 10 Continuity Test (CL21)                    1
023*  808 Channel 11 Continuity Test (CL21)                    1
024*  808 Channel 12 Continuity Test (CL21)                    1
025*  808 Channel 13 Continuity Test (CL21)                    2
026*  808 Channel 14 Continuity Test (CL21)                    1
027*  808 Channel 15 Continuity Test (CL21)                    1
030*  808 Channel 16 Continuity Test (CL21)                    1
031*  808 Channel 17 Continuity Test (CL21)                    1
032*  808 Channel 18 Continuity Test (CL21)                    1
033*  808 Channel 19 Continuity Test (CL21)                    1
034*  808 Channel 20 Continuity Test (CL21)                    1
035*  808 Channel 21 Continuity Test (CL21)                    1
036*  808 Channel 22 Continuity Test (CL21)                    1
037*  808 Channel 23 Continuity Test (CL21)                    1
```

Miscellaneous Register Tests

```
040   MC680 Register Test (CL06)                               3
041   MC6R1 Register Test (CL08)                               2
042   MC6R2 Register Test, Part 1 (CL08)                       3
043   MC6R2 Register Test, Part 2 (CL08)                       1
```

Miscellaneous Console Functions Tests

```
044*  CL09 DC Low Test (CL20)                                 2
045*  CL 808 Request Test (CL09)                               2
046*  EM83 CPU AC Low Test (CL08)                              1
047*  CL09 System AC Fail Test (CL09)                          4
050*  CL CPU Power Fail Interrupt Test (CL09)                  2
051*  CL Ubang Reset Test (CL18)                               2
052*  CL Master Reset Test (CL09)                              2
053*  CL ABus Enable Test (CL09)                               2
054*  CL Array DC OK Test (CL09)                               2
055*  CL09 CPU Alive Test (CL00)                               1
056*  CL09 Error 0 Test (CL08)                                 2
057*  CL09 Error 1 Test (CL08)                                 2
060*  CL09 Error 2 Test (CL08)                                 2
061*  CL09 CPU Control Store PE Test (CL08)                    6
062*  CL09 ABus Request 0 Test (CL08)                          2
063*  CL09 ABus Request 1 Test (CL08)                          2
064*  CL09 ABus Request 2 Test (CL08)                          2
065*  CL09 ABus Request 3 Test (CL08)                          2
066*  CL09 ABus Dead Interrupt Test (CL08)                     5
067*  SI30 Register Test (CL12)                                2
070*  SI01 Socket0 Test (CL12)                                 2
071*  SI02 Register Test (CL12)                                2
072*  CL IE0 Bit Test (CL08)                                   2
073*  CL IE1 Bit Test (CL08)                                   2
074*  CL IE2 Bit Test (CL08)                                   2
```

```
DIAGNOSTICS
CONSOLE MODULE DIAGNOSTIC (SDMBA)

KMN PCI Tests

075    KMN PCI Mode Register 1 Test (CL10)                    3
076    KMN PCI Mode Register 2 Test (CL10)                    3
077    KMN PCI Command and Status Register Tests (CL10)       4
100    KMN PCI Register Addressing Tests (CL10)               3
101    KMN PCI TXRDY Bit Test (CL10)                          1
102    KMN PCI Local Loopback Test, 19.2K Baud (CL10)         1
103    KMN PCI Pre Loopback Test, 19.2K Baud (CL10)           1
104    KMN PCI Bus Master Test (256 Transactions) (CL10)      1

Remote PCI Tests, Part 1

105    Remote PCI Mode Register 1 Test (CL10)                 3
106    Remote PCI Mode Register 2 Test (CL10)                 3
107    Remote PCI Command and Status Register Tests (CL10)    4
110    Remote PCI Register Addressing Tests (CL10)            3

Remote PCI Tests, Part 2

111*   LEXES Test (CL11)                                      2
112*   CL11 RTSRN Carrier Test (CL10)                         2
113*   CL11 RTSRN OSP Test (CL10)                             2
114*   CRIS Test (CL10)                                       2
115*   Assert CDIS Test (CL10)                                2
116*   REMOTE PCI TXRMT/RSCIN Test (CL10)                     3
117*   CL RTSRN OSC Test (CL10)                               1
120*   REMOTE PCI TXRDY Test, Part 1 (CL10)                   1

Remo PCI Tests, Part 3

121    Remote PCI Local Loopback Test,  75 Baud (CL10)        1
122    Remote PCI Local Loopback Test, 110 Baud (CL10)        1
123    Remote PCI Local Loopback Test, 150 Baud (CL10)        1
124    Remote PCI Local Loopback Test, 1800 Baud (CL10)       1

Remote PCI Test, Part 4

125*     Remote PCI Split Baud Rate Test (CL10)               1

SCP Interface Tests, Part 1

126    SCP Channel Continuity Test (SCP1)                     1
127    SCP Channel Init Test (SCP1)                           1
130    SCP CRI Drive/Sense Test (SCP1)                        1

SCP Interface Tests, Part 2

131*   SCP Switch Input Test (SCP1)                           2

CBUS Tests, Part 1

132    CL15 TX&L Test (CL14)                                  3
133*   CL15 TXCS RDY Test, Part 1 (CL13)                      3
134*   CL15 RXCS DNC Test, Part 1 (CL13)                      3
135*   CL15 STOR RDY Test, Part 1 (CL13)                      3
136    CL15 CSURT Test (CL13)                                 3
```

CBUS Tests, Part 2

139*	CBus RAM Data Test, TTL Port, Part 1 (CL16)	2
140*	CBus RAM Addressing Test, TTL Port (CL16/CL17)	1
141*	CBus Access Test, ECL Port (CL16)	2
142*	CBus RAM Addressing Test, ECL Port (CL16/CL17)	1
143*	CL15 TXCR IE Test (CL13)	5
144*	CL15 RXCR IE Test (CL13)	5
145*	CL15 STCR SIE Test (CL15)	6
146*	CL13 TBIXI Test (CL13)	3
147*	CL15 TXCR RDY Test, Part 2 (CL13)	2
150*	CL15 TXCR RDY Test, Part 3 (CL13)	2
151*	CL15 RXCR DNE Test, Part 2 (CL13)	2
152*	CL15 RXCR DNE Test, Part 3 (CL13)	1
153*	CL15 STCR RDY Test, Test 2 (CL13)	1
154*	CL15 STCR RDY Test, Part 3 (CL13)	1
155*	CL15 TSEL Command by SCC Port Test (CL13)	2
156*	CL TTX Interrupt Test (CL15)	4
157*	CL VRX Interrupt Test (CL15)	4
160*	CL RL02 Interrupt Test (CL15)	1

TOY Clock tests

161	TOY Software-Master-Reset Test (CL22)	16
162	TOY Counter Group 1 Register Test (CL22)	8
163	TOY Counter Group 2 Register Test (CL22)	8
164	TOY Counter Group 3 Register Test (CL22)	8
165	TOY Counter Group 4 Register Test (CL22)	8
166	TOY Counter Group 5 Register Test (CL22)	8
167	TOY Master Mode Register Test (CL22)	2
170	TOY Counter Register Count Test (CL22)	10
171	CL22 TOY 1S Out Test (CL22)	1

RAM Parity Tests

172	RAM Data Parity Tests (LP06)	3

Mapped RAM Data and Addressing Tests (64 TO 256 Kbytes)

173	Mapped RAM Data/Addressing Test (CL04)	2

I-11 Interrupt Tests

174	CL19 ENA TXRDY Test (CL20)	1
175	CL19 ENA STCR RDY Test (CL20)	1
176	Unsolicited Interrupt Test (CL02)	2
177	CL11 TSTRT Interrupt Test (CL02)	3
200	CL04 CSL DE Internal Interrupt Test (CL02)	6
201*	CL09 System AC Fault Interrupt Test, Part 1 (CL02)	5
202	CL09 ABus Load Internal Interrupt Test (CL02)	4
203	CL22 TOY 1KH Internal Interrupt Test (CL02)	6
204	CL15 TXCR RDY Interrupt Test (CL02)	5
205	CL15 STCR RDY Interrupt Test (CL02)	5
206	CL31 QBA Internal Interrupt Test (CL02)	6
207	CL10 RTCRN RDY Interrupt Test (CL02)	7
210	CL11 RVE8H Internal Interrupt Test (CL02)	6
211	CL10 Local RDY Interrupt Test (CL02)	5
212	CL30 RPLY Timeout Interrupt Test (CL02)	9

T-11 Interrupt Tests

213*	CL09 CPU Control Store PE Interrupt Test (CL02)	4
214*	T-11 Manual Restart Interrupt Test (CL02)	1

QBUS Adapter Tests

213	QCSR0 Register Test (CL19)	4
216	QCSR1 Register Test (CL26)	3
217	QBA Address Register Test (CL27)	2
220	QBA Data Register Test (CL27)	2
221	CL31 (SL Master Port Test (CL29)	5
222	QBA Timeout Counter Test (CL28)	6
223	CL30 REPLY Timeout Test, Part 1 (CL20)	4
224	QBA Simulated-Read-Cycle Test, Part 1 (CL23)	14
225	QBA Simulated-Read-Cycle Test, Part 2 (CL23)	2
226	QBA Simulated-Write-Cycle Test (CL23)	13
229	QBA Simulated-Interlock-Cycle Test, Part 1 (CL27)	8
230	QBA Simulated-Interlock-Cycle Test, Part 2 (CL23)	3
231	CL32 Event Bit Test (CL29)	3
232	QBA Simulated-DMA-Read Test, Part 1 (CL23)	24
233	QBA Simulated-DMA-Read Test, Part 2 (CL23)	3
234	QBA Simulated-DMA-Write Test (CL23)	24
235	CL30 REPLY Timeout Test, Part 2 (CL29)	2

RL02-specific DMA Tests

236	RL01 Register Read Test (CL23)	10
237	RL02 Register Write Test (CL25)	10
240	RL02 Register Write/Read Test (CL23)	2
241	RL02 DMA Read Test (CL29)	3

* Indicates a special test fixture is required. The test is ignored unless the fixture is installed.

6.4 EVKAA - MICRO-HARD-CORE DIAGNOSTIC (MHC)

OVERVIEW - EVKAA (MHC) is a T-11 based diagnostic. It is designed to verify the proper operation of the VAX CPU hardware logic required to successfully load and run micro-diagnostics.

6.4.1 Prerequisites

Successful execution of XXDRA. EVKAA also assumes that the CDP868.DAT or CDP865.DAT file on the RL02 is correct. CDP868.DAT and CDP865.dat contains the names and revision levels of the SDS signal name files.

6.4.2 Loading and Starting MHC

To load and start MHC type:

```
MHC(cr)
```

at either the Macro Context Prompt: >>>, or the Diagnostic Context Prompt: DC>. Once MHC has been started, it will display the version name and number, followed by the MHC prompt: MH>. MHC then waits for the user to type a command. See Table 6-5.

6.4.3 Micro-Bare-Core Control Characters

Character	Function
^C	Control C Each and test may be exited by typing ^C. The program will abort at the completion of the current sub test and return to the Micro-Hard-Core Prompt: MH>.
^T	Control T If the "/Performance" switch was selected, typing ^T will report the last sub test name at completion of the current sub test.

6.4.4 Micro-Hard-Core Control Switches

The user may append the following switches to a command in order to change the characteristics of the program execution.

SWITCH	FUNCTION
/Bell:	START0/BELL:arg(cr) Control the action of the Bell as specified by the argument. Valid arguments are: OFF (default) - Do not sound bell on error. ON - Sound bell on error.
/ERROR_DUMPS:	START0/ERROR_DUMPS:arg_#(CR) Report only the number Errors (per sub test) specified by the argument_#.

/FAULT: START/FAULT:arg<cr>
Upon detection of a error take the action specified by the argument. Valid arguments are:

CONTINUE - (Default) Continue on error.
ISOLATE - (default) Isolate the cause of the error to RAM Chip.
LOOP - Loop on failing test.
PAUSE - Pause on error and exit to "MH>". See CONTINUE Command.

/PASS: START//PASS:arg_#<CR>
Run selected tests for the number of passes specified by argument #.

/PRINT_MODE: START/PRINT_MODE:arg<cr>
Control the error reporting as specified by the argument. Valid arguments are:

BRIEF - Cancel "/PRINT_MODE:QUIET" and report errors in short form, no RAM chip callouts.
QUIET - Suppress all but error typeouts.
VERBOSE - (Default) Cancel "/PRINT_MODE:QUIET" and report errors in long form (i.e., RAM chip callouts)

/QUICKVERIFY START0/QUICKVERIFY<CR>
Run selected test with Quick Verify Flag set.

/NUMBER START/NUMBER:n1,n2<cr>
Run only tests "n1" thru "n2".

6.4.3 Micro-Hard-Core Commands

The following Table describes the commands associated with the Micro-Hard-Core Diagnostic.

Table 6-5 Micro-Hard-Core Commands

COMMAND	DESCRIPTION
START	Start<cr> Run one complete pass of all tests and then return to the Micro-Hard-Core Prompt (MH>).
STARTQ	START0<cr> Run one pass of all tests in Quick Verify Mode (shorter run time). QUICK VERIFY MODE: When the Quick Verify suffix is appended to either the START or LOOP command only the RAM addresses that cross physical chip boundaries are checked. For example, each bit of the RBox Control Store is 4192 addresses deep. The memory chip used has 1024 addresses.

When the START0 or LOOP0 Command is specified the program will not increment through each address. Instead it will update DPC <02:00>, which will select a bank of RAMs. Each bank represents 1K keywords of the total 8K. The DPC bits will be incremented until all 8 banks have been selected at least once.

The START0 and LOOP0 commands also executes certain RUN tests once instead of the default number of times. (See list below.) These two differences reduced the amount of time required to run the Micro-Hard-Core Diagnostic. In addition, however, they also reduce the fault detection level. In other words, you cannot be sure the RAMs are functioning properly unless you run at least one full pass of MHC.

GROUP TESTS

PBox	P-8, P-9, P-A, P-B, P-C, and P-F
IBox	I-A, I-B, and I-D
MBox	M-5 and M-6
EBox	E-C and E-D

START0 START0<cr>
Run Clock Tests only. See Table 6-6

STARTD STARTD<cr>
Run Disk Tests only. See Table 6-7.

STARTP STARTP<cr>
Run PBox Tests only. See Table 6-8.

STARTI STARTI<cr>
Run IBox Tests only. See Table 6-9.

STARTL STARTL<cr>
Run Last Multi Box Access Tests only. See Table 6-10.

STARTM STARTM<cr>
Run MBox Logic Tests only. See Table 6-11.

STARTW STARTW<cr>
Run MBox Write Tests only. See Table 6-12.

STARTE STARTE<cr>
Run EBox ECF-CTX RAM and Basic UCODE Tests only. See Table 6-13.

STARTS STARTS<cr>
Run XBox SBus and CS Tests only. See Table 6-14.

STARTU STARTU<cr>
Run SBox Expanded UCODE Tests only. See Table 6-15.

LOOP LOOP<cr>
Loop on all tests continuously until stopped by "^C".

LOOPQ LOOPQ<cr>
Continuously loop on all tests in Quick Verify Mode (shorter run time). See QUICK VERIFY MODE under START0.

LOOPC LOOPC<cr>
LOOP on Clock Tests only. See Table 6-6.

LOOPD	LOOPD<cr> Loop on Disk Tests only. See Table 6-7.
LOOPE	LOOPE<cr> Loop on EBox Tests only. See Table 6-8.
LOOPI	LOOPI<cr> Loop on IBox Tests only. See Table 6-9.
LOOPL	LOOPL<cr> Loop on Last Will Pox Access Tests only. See Table 6-10.
LOOPM	LOOPM<cr> Loop on MBox Logic Tests only. See Table 6-11.
LOOPK	LOOPK<cr> Loop on MBox LCHIP Tests only. See Table 6-12.
LOOPU	LOOPU<cr> Loop on CBox KDF-11A SAM and Basic UCODE Tests only. See Table 6-13.
LOOPS	LOOPS<cr> Loop on EBox SSR and CS Tests only. See Table 6-14.
LOOPU	LOOPU<cr> Loop on XBox Expanded UCODE Tests only. See Table 6-15.
CONTINUE	CONTINUE<cr> Continue on to the next sub test. This command should only be necessary if the "/DEFAULT:PAUSE" switch was selected and an error was detected.
REPRINT	REPRINT<cr> Retype the end of pass report. See Example 6-1

6.4.4 Micro-Hard-Core Test Descriptions

The following set of Tables list the subtests that can be run via the Micro-Hard-Core Diagnostic.

Table 6-6 Clock Subtests

TEST	MODULE AND DESCRIPTION
CLK (C-1)	LO217 (CLK) SSC SHIFT
CLK (C-2)	LO217 (CLK) SSC SHIFT CHAIN LOAD FUNCTION
CLK (C-3)	LO217 (CLK) VIS MUX SELECTS
CLK (C-4)	LO217 (CLK) VIS DATA AND LD_FUNC_REG_B
CLK (C-5)	LO217 (CLK) LOAD FREQUENCY REG
CLK (C-6)	LO217 (CLK) CLK2_START_H CIRCUIT
CLK (C-7)	LO217 (CLK) CLK3 MARK BIT
CLK (C-8)	LO217 (CLK) BUSY COUNTER READ AND WRITE
CLK (C-9)	LO217 (CLK) CPU CLOCK STOP
CLK (C-A)	LO217 (CLK) MARK BIT STOP CONDITION
CLK (C-B)	LO217 (CLK) FORCE COUNTER ABILITY TO COUNT
CLK (C-C)	LO217 (CLK) MBUS AND EBOX II TO IO SHIFT LOOP
CLK (C-D)	LO217 (CLK) MBUS T2 CLK-ENB,FEB,EMA,IOP MBUS ENABLE
CLK (C-E)	LO217 (CLK) STOP EBOX IN PHASE 0 AND PHASE 1

Table 6-7 RL02 Disk Subtests

TEST	MODULE and DESCRIPTION
D6K (D-1)	Disk (non-destructive) Write-Read Exerciser
DSK (D-A)	Disk Oscillation seek 1 = 1-1
DSK (D-B)	Disk Oscillation seek 1 = 1-89
DSK (D-C)	Disk Oscillation seek 1 = 1-178
DSK (D-D)	Disk Oscillation seek 1 = 1-345
DSK (D-E)	Disk Oscillation seek 1 = 1-511
DSK (D-F)	Disk (DESTRUCTIVE) Write-Read Exerciser

Table 6-8 FPU Subtests

TEST	MODULE and DESCRIPTION
FBOX (F-1)	L0213 (FBM) F BOX Present Verify
FBOX (F-2)	L0212 (FBA) SIN/COS/TAN AND LOOPBACK
FBOX (F-3)	L0211 (FBM) SEB SHIFT and LOOPBACK
FBOX (F-4)	FBOX Validate DPU Shift Like and ENABLE
FBOX (F-5)	L0217 (CLK) CLK LC RESET
FBOX (F-6)	L0212 (FBA) CPU REGISTER TESTS
FBOX (F-7)	L0213 (FBM) CPU REGISTER TESTS
FBOX (F-8)	L0212 (FBA) CONTROL STORE "55" DATA TEST
FBOX (F-9)	L0212 (FBA) CONTROL STORE "AA" DATA TEST
FBOX (F-A)	L0213 (FBM) CONTROL STORE "55" DATA TEST
FBOX (F-B)	L0213 (FBM) CONTROL STORE "AA" DATA TEST
FBOX (F-C)	L0213 (FBM) CONTROL STORE ADDRESS TEST
FBOX (F-D)	L0212 (FBA) CONTROL STORE ADDRESS TEST
FBOX (F-E)	L0213 (FBM) DECODE ROM "55" DATA TEST
FBOX (F-F)	L0213 (FBM) DECODE ROM "AA" DATA TEST
FBOX (F-10)	L0213 (FBM) OPCODE ADDRESS TEST
FBOX (F-11)	L0212 (FBA) BASIC M CPU-CODE TEST (EDKAF1)
FBOX (F-12)	L0212 (FBA) BASIC MICRO-CODE TEST (EJKAA1)
FBOX (F-13)	L0211 (FBM) EXPANDED CPU OPCODE TEST (EDKAF2)
FBOX (F-14)	L0212 (FBA) EXPANDED CPU OPCODE TEST (EDKAF2)
FBOX (F-15)	L0212 (FBA) MICRO-CODE TEST (EDKAB3)

Table 6-9 IBUS Subtests

TEST	MODULE and DESCRIPTION
IBUS (I-1)	SDB SHIFT AND LOOPBACK
IBOX (I-2)	SDB SELECT LINE AND ENABLE
IBUS (I-3)	L0200 (IBD) MODULE SDB CONTROL CHANNEL
IBOX (I-4)	L0207 (ICA) MODULE SDB CONTROL CHANNEL
IBUS (I-5)	L0217 (CLK) CLK LC RESET
IBOX (I-6)	L0207 (ICA) ICS CPU REGISTER TEST
IBOX (I-7)	L0207 (ICA) ICS "55" DATA TEST
IBOX (I-8)	L0207 (ICA) ICS "AA" DATA TEST
IBOX (I-9)	L0207 (ICA) ICS ADDRESS TEST
IBOX (I-A)	L0206 (IBD) IDRAM "55" DATA TEST
IBOX (I-B)	L0206 (IBD) IDRAM "AA" DATA TEST
IBOX (I-C)	L0206 (IBD) IDRAM ADDRESS TEST
IBOX (I-D)	L0707 (ICA) ICS RAM PARITY TEST
IBOX (I-E)	QUIESCENT STATE FOR I BOX CONTROL SIGNALS
IBUS (I-F)	L0207 (ICA) CPUCP MICRO-CODE TEST (EDKAI1)

NOTE

When executed on VAX-Hosa CPU, L0211 will be superseded in place of L0217.

Table 6-10 Left Box Subtests

TEST	MODULE and DESCRIPTION
VAX-8600	(L-1) CLK and K,I,N buses CMODE MARK R1Y
VAX-8500	(L-2) V2 SBI MASTER ESSEY CLK1
VAX-8650	(L-1) CLK and K,I,N buses CMODE MARK R1Y
VAX-8650	(L-2) V2 SBI MASTER ESSEY CLK1

NOTE

1. When executed on VAX-8650 CPU, L0231 will be reported in place of L0217

2. When executed on VAX-8650 CPU, L0230 will be reported in place of L0220

Table 6-11 MBox Logic Subtests

TEST	MODULE and DESCRIPTION
MBOX (M-1)	MBR SHIFT AND LOOPBACK
MIOX (M-2)	L0222 (MIOX) GPO SHIFT AND LOOPBACK
MESX (M-3)	SBI SILO/C LINK AND ENABLE
MDOX (M-4)	L0217 (CLK) CLK 141 RESET
MBOX (M-5)	L0220 (MCC) MCS "55" DATA TEST
MBOX (M-6)	L0220 (MCC) MCS "AA" DATA TEST
MBOX (M-7)	L0220 (MCC) CYCLE RAM "55" DATA TEST
MBOX (M-8)	L0220 (MCC) CYCLE RAM "AA" DATA TEST
MBOX (M-9)	L0220 (MCC) ACCESS RAM DATA TEST
MBOX (M-A)	L0220 (MCC) CYCLE RAM ADDRESS TEST
MBOX (M-B)	L0220 (MCC) MCE RAM ADDRESS TEST
MBOX (M-C)	L0220 (MCC) ACCESS RAM ADDRESS TEST

Table 6-12 EBox Ucode Subtests

TEST	MODULE and DESCRIPTION
EBOX (E-1)	L0220 (ECC) BASIC UCODE FUNCTION (EDKAN1)
EBOX (E-2)	L0220 (ECC) BASIC STACK MODE (EDKAP2)
EBOX (E-3)	L0220 (ECC) STACK PUSH LEVEL (EDKAM3)
EBOX (E-4)	L0220 (ECC) STACK POP LEVEL (EDKAM4)

Table 6-13 MCE, CTX, and MISC EBox Subtests

TEST	MODULE and DESCRIPTION
EBOX (E-1)	L0216 (CSE) DECHIPC SAVE INITILIZATION
EBOX (E-2)	L0216/L0216 (ECS) "55" 1 WORD (VSTARK) TEST
EBOX (E-3)	L0216/L0216 (ECS) "AA" 1 WORD (VSTERK) TEST
EBOX (E-4)	L0210 (EBC) MCE RAM DATA TEST
EBOX (E-5)	L0210 (EBC) MCE RAM ADDRESS TEST
EBOX (E-6)	L0210 (EBC) CONTEXT RAM DATA TEST
EBOX (E-7)	L0210 (EBC) CONTEXT RAM ADDRESS TEST
EBOX (E-8)	L0216/L0216 (ECS) SAV PARITY
EBOX (E-9)	L0210 (EBC) MCE RAM PARITY TEST
EBOX (E-A)	CHECKSUM STACK FOR CTRAP + STALL (EDBAC2)

```
EBOX [8-B]  L0216 [CSB] BASIC CPU OPCODE TEST (EDKAU1)
EBOX [8-C]  L0216 [CSB] EXTENDED CPU OPCODE TEST (EDKAU1)
EBOX [8-D]  L0216 [CSB] UJUMP REGISTER TEST (EDKAU2)
```

NOTE

When executed on VAX-8650 CPU, L0231 will be reported in place of L0217

Table 6-14 EBox, SDB, and C/S Subtests

TEST	MODULE and DESCRIPTION
FBOX [8-1]	MODULE: SDB SHIFT AND FLOATBACK
EBOX [8-2]	L0215 (CSA) MODULE SDB SHIFT AND LOOPBACK
EBOX [8-3]	SDB SELECT LINE AND ENABLE
EBOX [8-4]	L0215 (CSA)/L0216 (CSB) SDB CONTROL CHANNEL
DBOX [8-5]	L0209 (EDP) MODULE SDB CONTROL CHANNEL
EBOX [8-6]	L0210 (EBC) MODULE SDB CONTROL CHANNEL
FBOX [8-7]	L0219 (EBE) MODULE SDB CONTROL CHANNEL
EBOX [8-8]	L0210 (EBC) MODULE ICR TIME BASE COUNTER
EBOX [8-9]	L0210 (EBC) MODULE DIAG & BIE REG TEST
EBOX [8-A]	L0210 (EBC) MODULE MCR RAM TEST
EBOX [8-B]	L0217 (CLOCK) CLK 141 RESET
DBOX [8-C]	L0215 (CSA)/L0216 (CSB) DCS "55" DATA TEST
EBOX [8-D]	L0215 (CSA)/L0216 (CSB) DCS "AA" DATA TEST
BBOX [8-E]	L0215 (CSA)/L0216 (CSB) DCS Addr (MSB&LSB)
MBOX [8-F]	L0215 (CSA)/L0216 (CSB) ACK Addr (MSB&LSB)

Table 6-15 EBox Ucode Subtests

TEST	MODULE and DESCRIPTION
EBOX [0-1]	L0216 (CSB) MICROSTACK TEST (EDKAU2)
MBOX [0-2]	L0209 (EDP)/L0210(MBC)MBUS REQ CTX (EDKAU2)
EBOX [0-3]	L0209 (EDP) SHIFT COUNTER FUNCTION (EDKAU2)
EBOX [0-4]	L0209/L0206/L0202/L0205 DATA PATH (EDKAU2)
EBOX [0-5]	L0216 (CSB)/L0209 (EDP) BRANCH COND. (EDKAU2)
EBOX [0-6]	L0209 (EDP) ALU FUNCTION (EDKAU2)
EBOX [0-7]	L0209 (EDP) SCRATCH PADS DATA (EDKAU2)
EBOX [0-8]	L0219(EBE)/L0206(CSL)-CRRA/CSL-INT (EDKAU2)
EBOX [0-9]	L0219 (EBE) RDCS REGISTER (31:27) (EDKAU2)

Below is an example of the end of pass report when an error occurred.

Example 6-1: End of Pass Report

```
SECTION  TOTAL  OF ERRORS

CLOCK       (SC) = 00000
S SDP/CS (ES) = 00000
F RAMS   (SR) = 00000
E UCODE  (SU) = 00000
I BOX    (EI) = 00000
M IOGIC  (SM) = 00000
N UCODE  (SN) = 00000
X BOX    (SX) = 00000
RL02 DSK (SD) = 00000
VAX-8$xx (SJ) = 00000

PASS COUNTER = 00001
# OF ERRORS = 00004, TOTAL # OF ERRORS = 00009
```

6.5 MICRO DIAGNOSTIC CONTROL PROGRAM (DCP)

6.5.1 Overview

DCP is a VAX based Diagnostic Utility program designed to control the running of the micro diagnostics. It is entered by typing the DIAGNOSE command at either the XACRO Prompt (>>>) or the MicroVAX-Cobra Prompt (PMB>). Once entered, DCP will initialize the system to the point where microdiagnostics can be loaded, executed, and monitored. The steps taken by this initialization procedure are as follows:

1. Take control of the TXCB CSR interrupt vector.
2. Submit the DCP initialization file, DCLOAD.LOG, internally in order to perform the following steps.
   - Stop the CPU clock and perform a master reset.
   - Load the Diagnostic Support Microcode (DSM) into the CPUx control store.
   - Load other control stores, as necessary, to initialize parity logic, etc.
   - Enable the console's external logic interrupt, using the SET EXTI ON command.
   - Purge the CPUs to begin executing at the DSM start address and start the CPU clocks.
3. Pass the default diagnostic switch settings to DSM (via the CPUs).
4. Display the diagnostic command prompt (DC>) and avail. commands.

NOTE

DCLOAD.LOG contains a DEBUG command that leaves the DSM command set enabled when it terminates, as evidenced by the DC>> prompt.

6.5.2 DCP Control Characters

Control Character	Description
<CTRL/P>	Interrupts the currently running micro- diagnostic and returns control to the user. This places the microdiagnostic in the 'paused' state, allowing the user to examine or modify the status of the diagnostic test environment, and then resume execution of the microdiagnostic.
<CTRL/C>	Aborts the current command, which may include the need to abort a currently running microdiagnostic, and returns control to the user.

<CTRL/P> Commands DCP to display information about the state of the currently running microdiagnostic without disturbing its execution.

6.3.3 DCP Command Summary

Command	Description
CLEAR DATA	CLEAR DATA<cr> Clear the table of pointers defined by the SET DATA command and should be used prior to redefining a new set of hex scratchpad locations for error reporting purposes.
CONTINUE	CONTINUE<cr> This command allows the currently loaded microdiagnostic to resume test execution after being paused by either a switch setting (/FAULT:PAUSE, /FAULT:ISOLATE) or <CTRL/P>. The microdiagnostic restarts at the test following the one that was being executed when the pause occurred. If /SINGLE SCOPE is in effect and CONTINUE command is issued, single stepping is stopped and the remaining tests are run. A CONTINUE command, issued prior to giving either a RUN or START command, will result in an error message.
DEPOSIT	DEPOSIT/switch hex_addr hex_data<cr> This command allows the user to modify the FPDX scratchpad RAM, the system cache, or the MBUS RAM. The 32-bit hex data is deposited into the hexadecimal location of the specified RAM. The CPU clock must be running. Valid Switches: /CACHE /SCRATCH /MBUS
EXAMINE	EXAMINE/switch hex addr<cr> This command allows the user to examine the contents of specific FPDX scratchpad locations, cache locations, or MBus locations. The CPU clock must be running. Valid switches: /CACHE /SCRATCH /MBUS
GENERATE	GENERATE file.ext<cr> The generate command is used to evoke the DCP generate/verify process for generation and validation of isolation data. This command is not for field use. It is used to support the in-house development of isolation algorithms. There are no switches associated with this command. The only option is the name of the command file responsible for loading the microdiagnostic to be generated.

RUN

RUN/switch file.ext<cr>
This command initiates the execution of a command file whose purpose is to load and start a microdiagnostic program. Execution begins with the first number specified in the /NUMBER switch and ends with either the last test in the group of microdiagnostic tests or with the last test specified in the /NUMBER switch.

NOTE

1. The CPU clock must be running for the RUN command to work.

2. Switches can be combined in any order provided that the command line does not exceed 80 characters.

3. Switches appended to the RUN command remain in effect until the command completes.

4. Refer to the SWITCH Command for a description of this switch.

Valid Switches: (SWITCH DEFAULTS are printed in upper case)

/AUTO:[ON], OFF]
 See Note 4.

/FAULT:[SHORT, noabort, long, pause, continue, ignore]
 See Note 4.

/LINES:[ON, OFF]
 See Note 4.

/NUMBER:[first test,[(speed),comma](LAST_TEST]
This switch is used to specify the starting and ending test. Numbers in hex. If the switch is omitted, the default values are used (default = 0,7F).

/PASSES:[decimal number (default = 100]
The decimal number indicates the number of test passes to execute before continuing to the next test or attempting isolation. If the /PASSES switch is not specified, the default value (100) will be used. A special case of /PASSES:0 modifies the sequencing of the tests so that a particular group of tests can be run indefinitely. If the user specifies a 0 pass count, DCP will run each of the tests indefinitely (from FIRST to LAST) until <CTRL/C> or <CTRL/E> interrupts or a fault is detected. When a fault is detected, the action will be governed by the current setting of the /FAULT switch.

/PRINT_MODE:[brief, VERBOSE]
 See Note 4.

/SINGLE_STEP
When single step is enabled, diagnostic execution is interrupted after completing the proper number of passes of an individual test. A message, along with the test number just completed, is displayed and the operator is prompted for input. The STEP command can be used to run the next test and is followed by another pause when it completes. The CONTINUE command can be used to clear this mode and resume full speed test execution.

SET DATA — SET DATA scratch_address data_name<cr>

where:

- "scratch address" is a hexadecimal address within the XSeg scratchpad RAM (0 to FF)
- "data name" is a 1-30 character string, to be associated with the specified scratch data

This command allows a symbolic name to be assigned to a location in the XSeg scratchpad RAM. When a microdiagnostic test detects a fault and DCP is in verbose printing mode, the "data name" and data taken from the associated "scratch address" was included in the error report.

Set Data commands are normally used in command files responsible for loading microdiagnostics and setting diagnostic control switches. A maximum of 16 SET DATA commands can be used at one time. The CLEAR DATA command should be used to initialize the SET DATA memory prior to defining set data information. The SHOW DATA command can be used to display the current settings of SET DATA, along with the current state of the scratch variables.

SET ISOLATION — SET ISOLATION/switch<cr>

This command provides a way to modify default parameters associated with isolation. Once the defaults are modified, they remain in effect until DCP is reloaded (i.e., DIAG command).

Valid Switches:

/PASSES:decimal number
The decimal number indicates the number of times each test will be executed before continuing to the next test or attempting isolation (default = 100). It is not recommended that this number be altered in the field, as it affects the confidence level of any isolation data that may follow. Factory settings are preferred. This number should always be greater than zero (0).

/ERROR_LIMIT:decimal_number
The decimal number indicates the maximum number of error printouts that will occur due to faults with different syndromes within a given test (default = 10). Lowering this number has the effect of reducing the amount of error printouts at the expense of throwing away what may be useful data for tracking down an intermittent. This number has a maximum value of 10.

SET NAME — SET NAME microdiagnostic_filename<cr>

This command is normally used within a command file. It specifies the name of the microdiagnostic currently being loaded. This information is included in the fault report and is used to locate other associated files with the same name (different extensions).

SET SWITCH — SET SWITCH/sw:[oh]<cr>

This command sequences the default switch settings that govern the behavior of DCP and DSD in the control and running of microdiagnostics. A description of each switch is provided below, stating the default setting of the switch when DIAGNOSTIC control is entered.

NOTES

1. Switches can be combined in any order provided that the command line does not exceed 70 characters.

2. Switches altered with this command retain the new setting until changed again with the SET SWITCH command, or until DCP is reloaded.

Valid Switches:

/BELL:[ON,OFF]
When "ON," this switch causes the terminal bell to ring each time a new fault is detected and reported.

/FAULT:[:ISOLATE,noabort,loop,pause,continue,ignore]
This switch controls the action DCP takes after a microdiagnostic test detects a fault. Only one of the above arguments may be specified; they are mutually exclusive.

ISOLATE - This setting, which is the default, directs DCP to attempt isolation on all solid faults encountered. After the isolation attempt is completed, DCP returns to its command prompt. The CONTINUE or STEP commands can be used to resume test execution.

NOABORT - This setting directs DCP to attempt isolation on all solid faults encountered. After the isolation attempt is completed, DCP will continue onto the next test, thus providing a way to continue with the diagnostic execution after isolating.

LOOP - Tests within a diagnostic are targeted to run /PASS:0 times. When an error is detected, the standard error report is displayed and then DCP is instructed to loop forever on the failing test. Only <CTRL/C> or <CTRL/E> can terminate the loop.

PAUSE - This setting causes DCP to return to the command prompt after reporting a test failure. The CONTINUE or STEP command can be used to resume test execution.

CONTINUE - With this switch in effect, DCP will perform all of its normal error reporting. After each error report, DCP will continue its normal test sequencing as if no error had occurred. Isolation will not take place.

TRYHARD - This is a special switch intended only for microcode use. It has no useful application in the field. It directs DCP never to signal DCP that an error has occurred. So hardware errors can ever be detected or reported.

/LINES:[0], OFF]
This switch is used by microcoders to debug isolation code. If the switch has been set to ON and DCP is processing isolation statements, a source line number will be displayed as the statement is processed. This line number matches the line number of the isolation source statement that is currently being processed.

/PRINT_MODE:(brief, VERBOSE)
This switch controls the level of detail included in an error report.

SHOW DATA — SHOW DATA<CR>
This command displays all symbolic names and associated data currently defined by the SET DATA command. The range of diagnostic test numbers, with the number of passes, is also displayed under this command for the user's information.

SHOW SWITCHES — SHOW SWITCHES<CR>
This command displays the current switch settings and the default switch settings. The default settings are either those settings that were in place when DCP was invoked or those that have been modified via the SET SWITCH command.

The current switch settings are normally the same as the default settings since local settings (i.e., appending switches to RUN or START) return to default when the command completes.

The current switches will differ from the defaults when they are examined from within another command. This could occur by examining the switches from the current state of a RUN or START command which uses the /NOWAIT qualifier switch. When proceeding from that point with the CONTINUE or STEP command, the current switch settings will remain in effect until the initial RUN or START command completion.

START START/switch<cr>
This command initiates the execution of a test or group of tests within an already loaded microdiagnostic program. Execution begins with the first number specified in the /KUKBSH switch. It ends with either the last test in the group of microdiagnostic tests or with the last test specified in the /KUKBSH switch.

NOTE

1. The CPU clock must be running for the START command to work.

2. Switches can be combined in any order provided that the command line does not exceed 80 characters.

3. Switches appended to the START command remain in effect until the command completes.

4. Refer to the SWITCH Command for a description of this switch.

5. Refer to the DCP Command for a description of this switch.

Valid Switches: (SWITCH DEFAULTS are printed in upper case)

/BELL:[ON, OFF]
See Note 4.

/FAULT:[ISOLATE, noabort, loop, abuse, continue, ignore]
See Note 4.

/LINES:[24, OFF]
See Note 4.

/KUKBSH:/HOLT_TEST [(<param>,<comma>)/ LAST_TEST]
See Note 5.

/PASSED:decimal_number
See Note 5.

/PAUSE_MODE:[brief, VERBOSE]
See Note 4.

/SINGLE_Step
See Note 5.

STEP STEP<cr>
This command causes the next test in the currently loaded microdiagnostic to be executed. The STEP command is invalid unless the microdiagnostics have been started. Once they have been started and there is a pause in the diagnostic execution [/FAULT:PAUSE, /FAULT:ISOLATE, or <CTRL/P>], the STEP command may be used. A STEP command issued prior to giving either a RUN or START command will result in an error message.

6.6 VAX 86XX MICRODIAGNOSTICS

Table 6-16 lists the microdiagnostics for the VAX 86XX CPU. With the exception of Micro-Ear-Core these diagnostics are designed to run under control of the Diagnostic Control Program (DCP).

Table 6-16 VAX 86XX Microdiagnostics

NAME	DESCRIPTION
EDKRA	Micro-Ear-Core
EDKBA	EBox Series
EDKCA	MBox Series, Part 1
EDK1A	MBox Array Sv/Mn Go Funtion Test
EDKDA	MBox Series, Part 2
EDKJA	Array Minimum Functionality Test
EDK2A	MBox Series, Part 3
EDKVA	MBox Series, Part 4
EDKGA	MBox Series, Part 5
EDKHA	MBox Series, Part 6
EDKOA	IBox Series, Part 1
EDKPA	IBox Series, Part 2
EDKQA	IBox Series, Part 3
EDKRA	IBox Series, Part 4
EDKSA	IBox Series, Part 5
EDKTA	IBox Series, Part 6
EDKUA	IBox Series, Part 7
EDKVA	IBox Series, Part 8
EDKWA	IBox Series, Part 9
EDK1A	FBox Series, Part 1
EDK2A	FBox Series, Part 2
EDK3A	FBox Series, Part 3
EDK4A	FBox Series, Part 4
EDK5A	FBox Series, Part 5
EDK6A	Array Series
EDK6A	CBIA Series, Part 1
EDK7A	CBIA Series, Part 2

6.7 STANDARD MICRODIAGNOSTIC COMMAND FILES

Table 6-17 lists some common Command Files that can be executed under the control of DCP.

Table 6-17 Standard Microdiagnostic Command Files

Command File	Description
TSTCPU.COM	@TSTCPU<cr> Run one pass of the Micro-Ear-Core Diagnostic then run one pass each of the individual Micro-diagnostics.
MICROS.COM	@MICROS<cr> Run one pass each of the individual Micro-diagnostics. Do not run the Micro-Ear-Core first.

Table 6-17 Standard Microdiagnostic Command Files (Cont.)

Command File	Description
QVX1##.COM	@QVX<##><CR> Run one pass of the Micro-Hard-Core diagnostic then run the test sequences that are necessary to test (Quick Verify) the CPU module specified by ##. The individual Quick Verify tests are listed by module tested in Table 6-18.
QVL<##>.COM	@QVL<##><CR> Run one pass of the test sequences that are necessary to test (Quick Verify) the CPU module specified by ##. This series of Quick Verify Command Files will not run MMC first. The individual Quick Verify are listed by module tested in Table 6-18.

Table 6-18: MMC/Module Quick Verify Command Files

COMMAND FILE	MODULE TESTED
QVX202.COM	L0202 (SSR)
QVX203.COM	L0203 (SRA)
QVX204.COM	L0204 (XCU)
QVX205.COM	L0205 (XAF)
QVX220.COM	L0220 (XCC)
QVX222.COM	L0222 (MTM)
QVX206.COM	L0206 (IDP)
QVX207.COM	L0207 (TCA)
QVX208.COM	L0208 (IBD)
QVX214.COM	L0214 (ICB)
QVX209.COM	L0209 (EDE)
QVX210.COM	L0210 (XRV)
QVX211.COM	L0211 (XRH)
QVX215.COM	L0215 (CSA)
QVX216.COM	L0216 (CSB)
QVX219.COM	L0219 (SSE)
QVX212.COM	L0212 (FBA)
QVX213.COM	L0213 (IVR)
QVX218.COM	L0218 (FJX)
QVX223.COM	L0223 (FTM)
QVX217.COM	L0217 (CLK)

Table 6-19 CPU Module Quick Verify Command Files

COMMAND FILE	MODULE TESTED
QVL202.COM	[illegible] (SRS)
QVL203.COM	L0203 (SUA)
QVL204.COM	L0204 (MLL)
QVL205.COM	L0205 (MLP)
QVL220.COM	L0220 (MCC)
QVL222.COM	L0220 (KIM)
QVL206.COM	[illegible] (IDC)
QVL207.COM	L0207 (TCP)
QVL208.COM	L0208 (IBD)
QVL214.COM	L0214 (ICM)
QVL209.COM	L0209 (QUP)
QVL210.COM	L0210 (SBC)
QVL211.COM	L0211 (TMS)
QVL215.COM	L0215 (CSA)
QVL216.COM	L0216 (CSM)
QVL219.COM	L0219 (SBB)
QVL212.COM	L0212 (DSA)
QVL213.COM	L0213 (FBM)
QVL218.COM	L0218 (IJK)
QVL217.COM	[illegible] (C,E)

6.8 DIAGNOSTIC SUPERVISOR (DSAA) OVERVIEW

The Diagnostic Supervisor (DSAA) is a utility program designed to simplify loading, running, and controlling the operation of the following types of diagnostics:

- LEVEL 3 - Diagnostics that are designed to run under the Diagnostic Supervisor in standalone mode only. Typically these are: Functional Level Peripheral Diagnostics, Expair Level Peripheral Diagnostics, and CPU Cluster Diagnostics.
- LEVEL 2 - Diagnostics that are designed to run under the Diagnostic Supervisor in either standalone or on-line mode. Typically these are: Bus Interaction Diagnostics and Formatter/Reliability Peripheral Diagnostics.
- LEVEL 2M - Diagnostics that are designed to run under the Diagnostic Supervisor in on-line mode only. Typically these are Diagnostic Control Programs such as File Maintenance and Update Utilities.

6.8.1 Diagnostic Supervisor (DSAA) Control

The operation of the Diagnostic Supervisor is effected by either Control Characters (Table 6-20) or Direct Commands entered at the Diagnostic Supervisor Prompt: DSD (Table 6-21).

Table 6-20 Diagnostic Supervisor Control Character Summary

Control Character	Description
^C	Interrupt the execution of the diagnostic currently running.
^S	Suspend terminal output.
^Q	Resume terminal output.

Table 6-21 Diagnostic Supervisor Command Summary

Command	Description
SET LOAD	SET LOAD device:(directory)<cr> Sets the default load directory to a device and directory where the diagnostics and other maintenance software is stored; e.g., DMA0:[SYSMAINT].
SHOW LOAD	SHO LOAD<cr> Displays the default device and directory. See the SET LOAD command.
LOAD	LOAD filename.extension<cr> Load but do not start the file specified. The default extension is .EXE.

ATTACH

ATTACH
dev-type link-name dev-name parameters<cr>
Define a test path based on the device type, the link name, the device name, and the parameters specified. Table 6-12 lists some common device types, links, and device names along with associated parameters.

NOTE

1. A device must be attached using BUS as the link name before it can itself be used as a link. For example,

   DS> ATTACH DW780 BUS DW0 3 4<cr>

   In this command DW0 is defined as a DW780 that has, in this case, a TR Level of 3 and a BR Level of 4. DW0 can now be used as a link to further define the test path. For example,

   DS> ATTACH V8100 DW0 VBA0 760440 400 5<cr>

   In this case VBA0 is defined as a V8100 that is connected to the CPU via DW0. In addition the parameters specify that the Unibus Control and Status Register (CSR) address is 760440, the Vector Address is 400, and the BR Level is 4 for VBA0.

2. The test path (device name) must be selected before the test path can be tested. See SELECT command.

SELECT

SELECT device-name<cr>
Add the device specified to the list of units to be tested. See ATTACH and DESELECT commands.

DESELECT

DESELECT device-name<cr>
Remove the unit specified from the list of units to be tested.

SHOW DEVICE

SHOW DEVICE device-name<cr>
Display the characteristics of the device name specified. If ALL is specified display the characteristics of all devices.

SHOW SELECTED

SHOW SELECTED<cr>
Display the characteristics of all selected devices.

START

START/SWITCH/SWITCH...<cr>
Initialize the system and begin execution of the diagnostic program currently in memory. Switches appended to the command change the characteristics the program execution as follows:

/SECTION: START/SECTION:argument<cr>
Do not run the default sections of the diagnostic. Instead run only the section specified by the argument. The specific sections are unique to each diagnostic. Consult the program document for further information.

/PASS: START/PASS:arg<cr>
Run the program for the specified number [arg] times. The default is 1. If 0 is specified run the program indefinitely.

/TEST: START/TEST:arg1:arg2<cr>
Begin running the diagnostic at the first test specified (arg1) and run the diagnostic through the last test specified (arg2).

/SUBTEST: START/TEST:arg1/SUBTEST:arg2<cr>
If /SUBTEST is specified as an argument to /TEST then run the diagnostic from the first test specified (arg1) through the subtest specified (arg2).

RUN

RUN file.ext/switch/switch<cr>
Initialize the system, then load and run the program specified in accordance with the switches specified. If no file extension is specified then default to .SV3. The applicable switches are described under the START command.

SUMMARY

SUMMARY<cr>
Display a statistical report describing the performance of the diagnostic to date. Normally you would ask for a summary report after a diagnostic has run, or you would type ^T to interrupt the execution of the diagnostic, ask for a summary report, and then type CONTINUE to continue running the diagnostic.

CONTINUE

CONTINUE<cr>
Continue execution of the diagnostic currently in memory.

ABORT

ABORT<cr>
Execute the clean up routine necessary to return the system to a known state and return to the diagnostic supervisor command with status 05%. This is the recommended method of aborting the execution a diagnostic. Normally you would want to do this after diagnostic execution has been interrupt either because of a breakpoint or because you typed ^C.

SET FLAGS

SET FLAGS arg,arg,...arg<cr>
Set the program control flags specified by the arguments.

Valid control flags are:

HALT — (DEFAULT:OFF) Stop diagnostic execution if an error is detected.

LOOP — (DEFAULT:OFF) Loop on the first test that detects an error. Typing ^C will break the loop and return to the Diagnostic Supervisor prompt (DS>).

BELL — (DEFAULT:OFF) Sound the bell each time an error is detected.

IE1 — (DEFAULT:OFF) Inhibit all messages except those forced by either the diagnostic program or the diagnostic supervisor.

IE2 — (DEFAULT:OFF) Inhibit all but the first three lines (Device Information) associated with each error message.

IE3 — (DEFAULT:OFF) Inhibit any extended information associated with each error message.

IEO — (DEFAULT:OFF) Inhibit the summary (statistical) report that is normally displayed when diagnostic execution is complete.

QUICK — (DEFAULT:OFF) Run the diagnostic in Quick Verify Mode. This will reduce the diagnostic run time. In many cases the diagnostic does this by aborting test algorithms and by reducing the number of passes per test.

TRACE — (DEFAULT:OFF) Repeat the header line (name) of each test as it is executed.

OPERATOR — (DEFAULT:ON) When set indicates that an operator is available should operator intervention be required.

	PROMPT (DEFAULT/NONE) Enter extended dialogue. Display all limits and defaults associated with any questions associated with an operator response.
	ALL (DEFAULT:OFF) Set all flags. Note: HALT will take precedence over the LOOP.
SET FLAGS DEFAULT	Set FLAGS DEFAULT<cr> Restore the Control Flags to their default status. OPERATOR and PROMPT set, all others cleared.
SHOW FLAGS	SHOW FLAGS<cr> Display the state of each flag listed under the SET FLAGS command.
CLEAR FLAGS	CLEAR FLAGS arg,arg,arg<cr> Clear the flags specified by the arguments. The flags and their effect on the execution of a diagnostic are described under the SET FLAGS command. Also see the SET FLAGS DEFAULT command.
SET EVENT FLAGS	SET EVENT FLAGS arg,arg<cr> Set the event flags specified by the arguments. The arguments which range from 1 through 31 are: 1 Directs VMS to log errors detected by the diagnostics in the System Event. 2 Directs VMS to execute the relay algorithms when an error is detected and reported by a diagnostic. 3-31 These are program specific event flags. Refer to the individual diagnostic documents for specific details. ALL Set all Event Flags.
CLEAR EVENT FLAGS	CLEAR EVENT FLAGS arg,arg<cr> Clear the Event Flags specified by the arguments. The flags and their effect on the execution of a diagnostic are described under the SET EVENT FLAGS command.
SHOW EVENT FLAGS	SHOW EVENT FLAGS<cr> Display the state of each Event Flag listed under the SET EVENT FLAGS command.
SET BASE	SET BASE address<cr> Set the base address to the value specified by address argument. Once the base address is set all address references will be offset by the value of the base address. For example, if the base address was set to 200 and the command EXAMINE 50 were entered, the Diagnostic Supervisor would display the contents of address 250.

SET BREAKPOINT

SET BREAKPOINT address<cr>
Insert a breakpoint at the address specified by the address argument. If the program attempts to execute the instruction in the specified address, interrupt execution and return to the Diagnostic Supervisor Command Prompt (DS>. See the CONTINUE command. Also see the LINUX MAKA<CR>"M" command.

NOTE

The Diagnostic Supervisor will support up to 15 independent breakpoints.

CLEAR BREAKPOINT

CLEAR BREAKPOINT address<cr>
Remove the breakpoint that is set at the address specified by the address argument.

ALL If the argument ALL is specified instead of an address then clear all breakpoints.

SHOW BREAKPOINT

SHOW BREAKPOINT<cr>
Display the address associated with all breakpoints that are currently in effect.

SET DEFAULT

SET DEFAULT arg1,arg2<cr>
Set the default qualifiers for the EXAMINE and DEPOSIT commands where argument 1 represents data length and argument 2 represents the radix. Valid data length arguments are:

BYTE 08 bits
WORD 16 bits
LONG 32 bits (default)

Valid Radix arguments are:

HEX Hexadecimal (default)
DEC Decimal
OCT Octal

EXAMINE

EXAMINE/arg1/arg2 address<cr>
Display the contents of the address specified in the data length specified by argument 1, and in the radix specified by argument 2.

Valid data length arguments are:

/B - Byte (08 bits)
/W - Word (16 bits)
/L - Longword (32 bits)

Valid radix arguments are:

/H - Hexadecimal (base 16)
/D - Decimal (base 10)
/O - Octal (base 8)
/A - ASCII bytes

If no arguments are specified then display the contents of the address using the values specified by the last SET DEFAULT command.

If the address is preceded by one of the following symbols then interpret the address in the radix indicated:

%Daddress - Decimal
%Haddress - Hexadecimal
%Oaddress - Octal

DEPOSIT — DEPOSIT/arg1/arg2 address data<cr>
Deposit the data specified in the address specified. If no arguments are specified then interpret data in the data length and radix values specified by the last SET DEFAULT command. The arguments are the same as shown for the EXAMINE command.

NEXT — NEXT arg<cr>
Execute the decimal number of macro instructions specified by the argument. Then display the new PC and the next 4 bytes.

NEXT

Normally this command would be used in conjunction with breakpoints.

Table 6-22 Device List for ATTACH, SELECT, and DESELECT Commands

Device Type	Link	Device Name	Additional Parameters
CR11	CRa	CRn	<unit><vector><ubr>
DMC11	DMa	DMan	<ucsr><vector><ubr>
DR11B	DRa	DRn	<ucsr><vector><ubr>
DU750	SBI	DTa	<tr>
DUP11	DUa	DUan	<ucsr><vector><ubr>
DW780	SBI	DWa	<tr>
DX11	DXa	DXn	<ucsr><vector><ubr><XTA> (<=DMA>
KA780	SBI	KAn	<0-floating><n-floating> <MCS-ipa>-address>
KMC11	KMa	KMan	<ucsr><vector><ubr>
LP11	LPa	LPn	<unit><vector><ubr>
MS780	SBI	MSa	<tr>
PCL11	PAa	PAn	<ucsr><vector><ubr>
UB780	SBI	UBa	<tr>
RK06	DPa	DMan	
RK07	DKa	DMan	
RX611	DXa	DXn	<unit><vector><ubr>
RL02	DYa	DYan	
RL11	DYa	DYn	<ucsr><vector><ubr>
RM03	DRa	DRan	
RP04	DBa	DBan	
RE05	DTa	DTan	
RP06	DBa	DBan	
RP07	DBa	DBan	
TE16	MTa	MTan	
TM03	TMa	MTa	<drive>
TS04	DWa	MTan	<ucsr><vector><ubr>
TU45	MTa	MTan	
TU77	MTa	MTan	

The definitions for the additional parameters are:

Parameter	Description	Radix	Range
<tr>	Adapter TR Number	Decimal	1-15
 	Adapter BR Level	Decimal	4-7
<drive>	Massbus Drive	Decimal	0-7
<csr>	Unibus CSR Address	Octal	760000-777776
<vector>	Unibus Vector	Octal	0-774
<ipl>	Unibus BR Level	Decimal	4-7

6.0 BL02 RESIDENT VAX MACRO-DIAGNOSTICS

Table 6-23 lists the Macro-Diagnostics that are available on the BL02 load medium.

Table 6-23: BL02 Resident VAX Macrodiagnostics

Name	Description
EVKAA.EXE	VAX Hardcore Instruction Test
EDBAA.EXE	VAX 8600 Diagnostic Supervisor (8600/8650)
EVKAB.EXE	VAX Basic Instruction Exerciser (8600/8650)
EVKAB.EXE	VAX Basic Instruction Exerciser
EVKAC.EXE	VAX Floating-Point Exerciser
EVKAD.EXE	VAX Compatibility Mode Instructions Exerciser
EVKAF.EXE	VAX Privileged Architecture Exerciser
EVCAA.EXE	BI780 Diagnostic
EVCBA.EXE	DW780 Repair Diagnostic
EDCLA.EXE	DW780, CI780, DW780, DW780 SBI Exerciser (8600/8650)
EVCGA.EXE	CI780 Repair Level Diagnostic Part 1
EVCGB.EXE	CI780 Repair Level Diagnostic Part 2
EVCGC.EXE	CI780 Repair Level Diagnostic Part 3
EVCGD.EXE	CI780 Repair Level Diagnostic Part 4
EVCAA.EXE	CI780 Functional Diagnostic Part 1
EVCAB.EXE	CI780 Functional Diagnostic Part 2
EVMAA.EXE	VAX Generic Tape Exerciser
EVMAB.EXE	TM03-TU16/TU45/TU77 Function Timing Tests
EVMAC.EXE	TM03-TU16/TU45/TU77 Control Logic Tests
EVMAK.EXE	VAX TM78/TU78 Control Logic Diagnostic
EVMBB.EXE	TU81 Front End/Host Functional Diagnostic
EVMBD.EXE	VAX TU80 Functional Diagnostic Part 1
EVMBE.EXE	VAX TU80 Functional Diagnostic Part 2
EVQTY.EXE	TK78 Loadable Driver
EVQTN.EXE	TM03-TU16/TU77 Loadable Driver
EVQTS.EXE	VAX TS11 Standalone Driver
EVQLE.EXE	VAX UBE Driver/UBE QIO Driver
EVRFA.EXE	VAX RFA and UA Drive Diagnostic
EVRLB.EXE	VAX RA60/RA80/RA81 Variables
EVSBA.EXE	Autosizer Level 3
EDKAK.EXE	VAX 8600 CPU Cluster Exerciser (8600/8650)

CHAPTER 7

SYSTEM INFORMATION

7.1 VAX 8600 TO 8650 DIFFERENCES/UPGRADE

The packaging, reliability, and features of the VAX 8650 are the same as the VAX 8600. The physical characteristics and footprint are the same. The VAX 8650 delivers 1.44 times the performance of the VAX 8600. The performance improvement has been achieved by increasing the clock rate from 50 MHz to 72 MHz, reducing the CPU microcycle time from 80 nsec to 55 nsec.

The increase in clock rate requires two new modules, and requires that certain other modules be at specified revisions as per the following table. Note that the MCC and CLK modules are new modules for the VAX 8650.

Modules Pertaining to VAX 8600/8650 Upgrade

Module Name	VAX 8600	VAX 8650	Revision
MFC--MBox Control	L0220	L0230	--
CLK--Clock	L0217	L0231	--
MDP--MBox Data Paths	L0204	L0304	E1
MAP--MBox Address Paths	L0205	L0305	E1
IDP--IBox Data Paths	L0206	L0306	B1
EBD--EBox Data Paths	L0211	L0211	E1
MBA--MBox Adder	L0212	L0212	B1
MOS Memory Array, 4 Mb	L0200	------	<b1
MOS Memory Array, 4 Mb	L0200	L0200	=,>D1
MOS Memory Array, 4 Mb	L0225	L0226	Any
MOS Memory Array, 16 Mb	L0225	L0225	Any
MOS Memory Array, 64 Mb	L0235	L0235	Any

NOTES

1. The L0230 (MFC) and L0231 (CLK) make up the 8610P-AA Upgrade Kit.

2. The revision shown for the L0204, L0205, L0206, L0211, and L0212 is the minimum revision required to upgrade from 8600 to 8650. If these modules are not up to minimum revision, the upgrade prerequisite kit, 8610P-BA, must be ordered.

3. The L0200, revision level D1 or above, L0226, L0225, and L0235 are backward compatible with the VAX 8600.

4. The L0200 below revision level D1 can only be used on the VAX 8600.

5. The minimum required version of VAX/VMS to support the VAX 8650 is Version 4.3.

Figure 7-1 System Block Diagram

Figure 7-2 CPU Block Diagram

7.2 KA8600/8650 RAM AND MICROCODE LISTING INFORMATION

7.2.1 KA8600/8650 Common RAM and Microcode Listing Information

Control Store	Last Addr	Size (DxW)	Data Used	List Name	Module Location	Console Label
MIC	1F	256x12	E	BCODE	SDP/209	ESC
Context	1FF	512x4	X	CVX	SBC/210	CONTEXT
FADR CS	1FF	512x40	R	FRFD	FBA/212	FRACS
FDRAM	3FF	1kx8	G	FALD	FBM/213	FDRAM
FRDL CS	1FF	512x40	F	FRDL	FBM/213	FDRCS
MCE	FF	256x16	V	MCP	SBC/210	MCE
IBox CS	1F	256x72	I	IeUA	I.A/207	IBS

7.2.2 KA8600 Specific RAM and Microcode Listing Information

Control Store	Last Addr	Size (DxW)	Data Used	List Name	Module Location	Console Label
DBox CS	1FFF	8kx92	I	KA86C0	CSA/215 CSB/216	BCS
IDRAM	FFF	4kx20	D	KA86C0	I3D/206	IDRAM
MBox CS	1FF	256x80	M	MCODE0	MCC/220	MCS
CYCLD	FF	256x20	K	CYCLE	MCC/220	CYCLE
Accmem	FF	256x	H	ACCMEM	MCC/220	ACCMEM

7.2.3 KA8650 Specific RAM and Microcode Listing Information

Control Store	Last Addr	Size (DxW)	Term Used	List Name	Module Location	Console Label
IBox CS	1FFF	8kx92	J	KA8650	CC4/215 CC8/216	ICS
IDRAM	1FF	8kx20	D	KA8650	IDD/200	IDRAM
MBox CS	FF	256x80	M	UCODE0	MCC/230	MCS
ACCMEM	1F	256x1	S	ACCMEM	MCC/230	ACCMEM
CYCLE	FF	256x20	N	CYCLE	MCC/230	CYCLE

Field	Description
Control Store:	The name of the control store.
Last address:	The last address within the range of the specified (entire) store.
Size (DxW):	Size of the control store: Depth (D, number of locations) X Width (W, number of bits).
Term Code:	The term used in the listings files which indicates the contents of the specified control store location.
Listing Name:	The listing name where the contents of the specified control store can be found. Listing has a .MCR extension.
Module Location:	The module mnemonic and module number where the control store is located.
Console Label:	The console mnemonic used with the EXAMINE and DEPOSIT commands. Note 1.

NOTES

1. With the exception of the ECC, the SDX debugger must be loaded and the clock stopped to deposit/examine these control stores.

2. The CONTEXT RAMs are 1K RAMs.

3. The file extension for the listing files is .MCR (e.g., EABD.MCR). The actual source files (for loading from the console disk to the control stores) have either a .BSN (binary packed normalized) or .PHY (physical/ASCII) extension.

4. During system initialization the console software reads EAMnn.REV (where n=0 for a KA620 and n=5 for a KA650). The contents of this file is used to ensure that the correct version of the microcode is loaded.

7.5 INTERRUPT LEVEL ASSIGNMENTS/SOURCES

Level	Condition	Vector	Source
IPR 1F	Mode Assigned	------	------
IPR 1E	CPU Power Fail	00C	Internal
	SBI 0 Fail	064	External
	SBI 1 Fail	364	External
IPR 1D	M Box Error	0b4	Internal
IPR 1C	SBIA 0 Error	060	External
	SBI 0 Fault	05C	External
	UNIB 1 Error	260	External
	SBI 1 Fault	25C	External
IPR 1B	SBI 0 Alert	058	External
	SBI 1 Alert	258	External
IPR 1A	None Assigned	------	------
IPR 19	SBI 0 Silo Compare . . .	050	External
	SBI 1 Silo Compare . . .	250	External
IPR 18	Interval Timer	0C0	Internal
IPR 17	SBI 0 REQ 7/UNIBUS BR 7 .	1c0-1fc	External
	SBI 1 RXD 7/UNIBUS BR 7 .	3C0-3FC	External
IPR 16	SBI 0 REQ 6/UNIBUS BR 6 .	180-1BC	External
	SBI 1 RXD 6/UNIBUS BR 6 .	380-3BC	External
IPR 15	SBI 0 REQ 5/UNIBUS BR 5 .	140-17C	External
	SBI 1 RXD 5/UNIBUS BR 5 .	340-37C	External
IPR 14	Console Terminal Receive	0F8	Internal
	Console Terminal Transmit	0FC	Internal
	SBI 0 RXD 4/UNIBUS BR 4 .	100-13C	External
	SBI 1 RXD 4/UNIBUS BR 4 .	300-33C	External
IPR 13	None Assigned	------	------
IPR 12	None Assigned	------	------
IPR 11	None Assigned	------	------
IPR 10	None Assigned	------	------
IPR 0F	Software Request 0F . . .	0BC	Software
IPR 0E	Software Request 0E . . .	0B8	Software
IPR 0D	Software Request 0D . . .	0B4	Software
IPR 0C	Software Request 0C . . .	0B0	Software
IPR 0B	Software Request 0B . . .	0AC	Software
IPR 0A	Software Request 0A . . .	0A8	Software
IPR 09	Software Request 09 . . .	0A4	Software
IPR 08	Software Request 08 . . .	0A0	Software
IPR 07	Software Request 07 . . .	09C	Software
IPR 06	Software Request 06 . . .	098	Software
IPR 05	Software Request 05 . . .	094	Software
IPR 04	Software Request 04 . . .	090	Software
IPR 03	Software Request 03 . . .	08C	Software
IPR 02	Software Request 02 . . .	050	Software
	or AST Delivery		
IPR 01	Software Request 01 . . .	084	Software

This table represents only those external interrupts which can originate from the two SBI adapters or SBI devices attached to them.

Generally, the ABus supports hardware interrupt levels 10 through 1F. Future ABus attachments may make use of this capability and require the assignment of additional locations in the SCB.

7.4 VAX 8600/8650 SYSTEM CONTROL BLOCK

Vector	Name	Type	Code
000	Unused	----	3
004	Machine Check	Fault/Abort	1
008	Kernel Stack Not Valid	Abort	1
00C	I/O Power Fail	Interrupt	1
010	BUG Reserved Opcodes & Privileged Instructions	Fault	0
014	Reserved Customer Opcodes	Fault	0
018	Reserved Operand	Fault/Abort	0
01C	Reserved Addressing Modes	Fault	0
020	Access Control Violation	Fault	0
024	Translation not Valid	Fault	0
028	Trace Pending	Fault	0
02C	Breakpoint	Fault	0
030	Compatibility Mode	Fault/Abort	0
034	Arithmetic	Trap/Fault	0
038	Unused	----	3
03C	Unused	----	3
040	CHMK Opcode	Trap	0
044	CHME Opcode	Trap	0
048	CHMS Opcode	Trap	0
04C	CHML Opcode	Trap	0
050	SBI 0 Silo Compare	Interrupt	1
054	Array Single Bit Error	Interrupt	1
058	SBI 0 Alert	Interrupt	1
05C	SBI 0 Fault	Interrupt	1
060	SBI 0 Error	Interrupt	1
064	SBI 0 Fail	Interrupt	1
068/06C	Unused	----	3
070	Software Request 01	Interrupt	1
074	Software Request 02 or AST Delivery	Interrupt	0
078	Software Request 03	Interrupt	1
07C	Software Request 04	Interrupt	1
080	Software Request 05	Interrupt	1
084	Software Request 06	Interrupt	1
088	Software Request 07	Interrupt	1
08C	Software Request 08	Interrupt	1
090	Software Request 09	Interrupt	1
094	Software Request 0A	Interrupt	1
098	Software Request 0B	Interrupt	1
09C	Software Request 0C	Interrupt	1
0A0	Software Request 0D	Interrupt	1
0A4	Software Request 0E	Interrupt	1
0A8	Software Request 0F	Interrupt	1
0AC	Interval Timer	Interrupt	1
0B0/0B4	Unused	----	3
0B8	Console Block Storage	Interrupt	0
0BC	Unused	----	3
0C0	Console Terminal Receive	Interrupt	1
0C4	Console Terminal Transmit	Interrupt	1
100-13C	SBI 0 REQ 4/UNIBUS BR 4	Interrupt	1
140-17C	SBI 0 REQ 5/UNIBUS BR 5	Interrupt	1
180-1BC	SBI 0 REQ 6/UNIBUS BR 6	Interrupt	1
1C0-1FC	SBI 0 REQ 7/UNIBUS BR 7	Interrupt	1

Vector	Name	Type	Code
200-240	Unused	-----	0
250	SBI 1 File Compare	Interrupt	1
254	SBI 1 Fail	Interrupt	1
258	SBI 1 Alert	Interrupt	1
25C	SBI 1 Fault	Interrupt	1
260	SBI 1 Error	Interrupt	1
264-2FC	Unused	-----	0
300-33C	SBI 1 REQ 4/CSIBUS BR 4	Interrupt	1
340-37C	SBI 1 REQ 5/CSIBUS BR 5	Interrupt	1
380-3BC	SBI 1 REQ 6/CSIBUS BR 6	Interrupt	1
3C0-3FC	SBI 1 REQ 7/CSIBUS BR 7	Interrupt	1
400-7FC	Unused	-----	0

Code	Description
0	Service on Kernel Stack (Interrupt Stack if selected)
1	Service on Interrupt Stack
2	Service via XCB (if no ACS HALT)
3	Reserved

NOTE

The Interrupt Vectors for SBI 1 are offset from SBI 0 by a page (200 words). The offset is added by the SBIA Interrupt handling microcode after it examines CSLIDI <22:21> and determines which adapter caused service.

7.5 CONFIGURATION PARAMETERS

7.5.1 Kernel System

The basic parts of the "Kernel" system are the CPU and the front end subset. These two parts make up the base level machine from which all system extensions are derived.

The kernel system consists of the following:

1. CPU cabinet (1)
2. DW86 SBI Adapter (1)
3. DB112 redwa (1for 48 Mb systems only (1))
4. DW780 UNIBUS adapter (1)
5. ROC MOS memory (up to 64 Megabytes)
6. Front End Subsets (FES) (1)

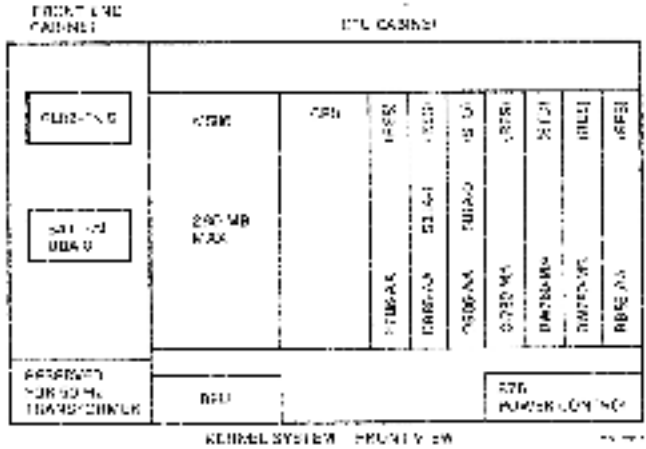


Figure 7-3 Kernel System - Front View

Nomenclature:

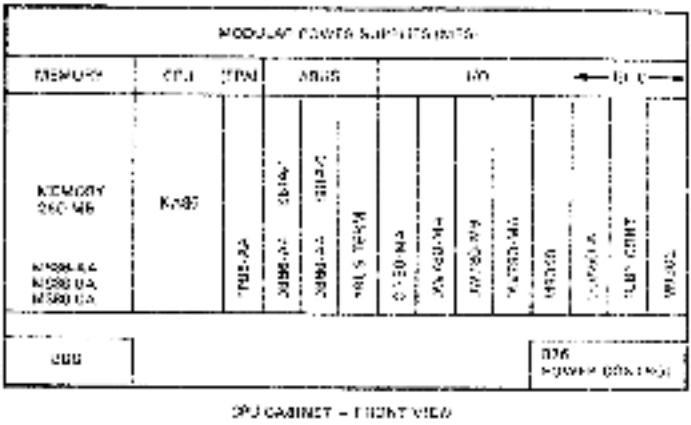
1.	CI780-AA	SBI-based Computer Interconnect Adapter
2.	DW56-AA	SBI Adapter (NI/A)
3.	DW780-AA	SBI-based UNIBUS Adapter
4.	FP86-AA	Floating Point Accelerator
5.	MS86-AA, BA, CA, or DA	ECC MOS Memory
6.	RB86-AA	IDVC option (comprised of DW780-AA and UDA50-A)
7.	RES	Reserved for expansion
8.	STD	Standard, part of basic KLUBAS package

7.5.2 CPU Cabinet Expansion

The CPU can accommodate the followings

1.	CI780-AA	Computer Interconnect (1)
2.	DW86-AA	SBI Adapter (2)
3.	DW780-AA, DW780-AA	UNIBUS adapters (2)
4.	FP86-AA	Floating Point Accelerator (1)
5.	MS86-AA, BA, CA, or DA	(up to 260 megabytes with battery backup)
6.	RB86-AA	IDVC option (1)

These options have dedicated and pre-wired slots within the CPU cabinet. Only the proper variations of these options can be configured into the CPU cabinet.


MODULAR POWER SUBSYSTEM
MEMORY
CPU
FPU
ABUS
I/O
I/O
MEMORY
256 MB
KA780
FPU-AA
CI780-AA
CI780-BA
MS780-AA
MS780-BA
MS780-CA
MS780-DA
KA780-AA
KA780-BA
KA780-CA
KA780-DA
KA780-EA
KA780-FA
KA780-GA
KA780-HA
KA780-JA
KA780-KA
KA780-LA
KA780-MA
KA780-NA
KA780-PA
KA780-QA
KA780-RA
KA780-SA
KA780-TA
KA780-UA
KA780-VA
KA780-WA
KA780-XA
KA780-YA
KA780-ZA
KA780-AB
KA780-BB
KA780-CB
KA780-DB
KA780-EB
KA780-FB
KA780-GB
KA780-HB
KA780-JB
KA780-KB
KA780-LB
KA780-MB
KA780-NB
KA780-PB
KA780-QB
KA780-RB
KA780-SB
KA780-TB
KA780-UB
KA780-VB
KA780-WB
KA780-XB
KA780-YB
KA780-ZB
KA780-AC
KA780-BC
KA780-CC
KA780-DC
KA780-EC
KA780-FC
KA780-GC
KA780-HC
KA780-JC
KA780-KC
KA780-LC
KA780-MC
KA780-NC
KA780-PC
KA780-QC
KA780-RC
KA780-SC
KA780-TC
KA780-UC
KA780-VC
KA780-WC
KA780-XC
KA780-YC
KA780-ZC
KA780-AD
KA780-BD
KA780-CD
KA780-DD
KA780-ED
KA780-FD
KA780-GD
KA780-HD
KA780-JD
KA780-KD
KA780-LD
KA780-MD
KA780-ND
KA780-PD
KA780-QD
KA780-RD
KA780-SD
KA780-TD
KA780-UD
KA780-VD
KA780-WD
KA780-XD
KA780-YD
KA780-ZD
KA780-AE
KA780-BE
KA780-CE
KA780-DE
KA780-EE
KA780-FE
KA780-GE
KA780-HE
KA780-JE
KA780-KE
KA780-LE
KA780-ME
KA780-NE
KA780-PE
KA780-QE
KA780-RE
KA780-SE
KA780-TE
KA780-UE
KA780-VE
KA780-WE
KA780-XE
KA780-YE
KA780-ZE
KA780-AF
KA780-BF
KA780-CF
KA780-DF
KA780-EF
KA780-FF
KA780-GF
KA780-HF
KA780-JF
KA780-KF
KA780-LF
KA780-MF
KA780-NF
KA780-PF
KA780-QF
KA780-RF
KA780-SF
KA780-TF
KA780-UF
KA780-VF
KA780-WF
KA780-XF
KA780-YF
KA780-ZF
KA780-AG
KA780-BG
KA780-CG
KA780-DG
KA780-EG
KA780-FG
KA780-GG
KA780-HG
KA780-JG
KA780-KG
KA780-LG
KA780-MG
KA780-NG
KA780-PG
KA780-QG
KA780-RG
KA780-SG
KA780-TG
KA780-UG
KA780-VG
KA780-WG
KA780-XG
KA780-YG
KA780-ZG
KA780-AH
KA780-BH
KA780-CH
KA780-DH
KA780-EH
KA780-FH
KA780-GH
KA780-HH
KA780-JH
KA780-KH
KA780-LH
KA780-MH
KA780-NH
KA780-PH
KA780-QH
KA780-RH
KA780-SH
KA780-TH
KA780-UH
KA780-VH
KA780-WH
KA780-XH
KA780-YH
KA780-ZH
KA780-AJ
KA780-BJ
KA780-CJ
KA780-DJ
KA780-EJ
KA780-FJ
KA780-GJ
KA780-HJ
KA780-JJ
KA780-KJ
KA780-LJ
KA780-MJ
KA780-NJ
KA780-PJ
KA780-QJ
KA780-RJ
KA780-SJ
KA780-TJ
KA780-UJ
KA780-VJ
KA780-WJ
KA780-XJ
KA780-YJ
KA780-ZJ
KA780-AK
KA780-BK
KA780-CK
KA780-DK
KA780-EK
KA780-FK
KA780-GK
KA780-HK
KA780-JK
KA780-KK
KA780-LK
KA780-MK
KA780-NK
KA780-PK
KA780-QK
KA780-RK
KA780-SK
KA780-TK
KA780-UK
KA780-VK
KA780-WK
KA780-XK
KA780-YK
KA780-ZK
KA780-AL
KA780-BL
KA780-CL
KA780-DL
KA780-EL
KA780-FL
KA780-GL
KA780-HL
KA780-JL
KA780-KL
KA780-LL
KA780-ML
KA780-NL
KA780-PL
KA780-QL
KA780-RL
KA780-SL
KA780-TL
KA780-UL
KA780-VL
KA780-WL
KA780-XL
KA780-YL
KA780-ZL
KA780-AM
KA780-BM
KA780-CM
KA780-DM
KA780-EM
KA780-FM
KA780-GM
KA780-HM
KA780-JM
KA780-KM
KA780-LM
KA780-MM
KA780-NM
KA780-PM
KA780-QM
KA780-RM
KA780-SM
KA780-TM
KA780-UM
KA780-VM
KA780-WM
KA780-XM
KA780-YM
KA780-ZM
KA780-AN
KA780-BN
KA780-CN
KA780-DN
KA780-EN
KA780-FN
KA780-GN
KA780-HN
KA780-JN
KA780-KN
KA780-LN
KA780-MN
KA780-NN
KA780-PN
KA780-QN
KA780-RN
KA780-SN
KA780-TN
KA780-UN
KA780-VN
KA780-WN
KA780-XN
KA780-YN
KA780-ZN
KA780-AP
KA780-BP
KA780-CP
KA780-DP
KA780-EP
KA780-FP
KA780-GP
KA780-HP
KA780-JP
KA780-KP
KA780-LP
KA780-MP
KA780-NP
KA780-PP
KA780-QP
KA780-RP
KA780-SP
KA780-TP
KA780-UP
KA780-VP
KA780-WP
KA780-XP
KA780-YP
KA780-ZP
KA780-AQ
KA780-BQ
KA780-CQ
KA780-DQ
KA780-EQ
KA780-FQ
KA780-GQ
KA780-HQ
KA780-JQ
KA780-KQ
KA780-LQ
KA780-MQ
KA780-NQ
KA780-PQ
KA780-QQ
KA780-RQ
KA780-SQ
KA780-TQ
KA780-UQ
KA780-VQ
KA780-WQ
KA780-XQ
KA780-YQ
KA780-ZQ
KA780-AR
KA780-BR
KA780-CR
KA780-DR
KA780-ER
KA780-FR
KA780-GR
KA780-HR
KA780-JR
KA780-KR
KA780-LR
KA780-MR
KA780-NR
KA780-PR
KA780-QR
KA780-RR
KA780-SR
KA780-TR
KA780-UR
KA780-VR
KA780-WR
KA780-XR
KA780-YR
KA780-ZR
KA780-AS
KA780-BS
KA780-CS
KA780-DS
KA780-ES
KA780-FS
KA780-GS
KA780-HS
KA780-JS
KA780-KS
KA780-LS
KA780-MS
KA780-NS
KA780-PS
KA780-QS
KA780-RS
KA780-SS
KA780-TS
KA780-US
KA780-VS
KA780-WS
KA780-XS
KA780-YS
KA780-ZS
KA780-AT
KA780-BT
KA780-CT
KA780-DT
KA780-ET
KA780-FT
KA780-GT
KA780-HT
KA780-JT
KA780-KT
KA780-LT
KA780-MT
KA780-NT
KA780-PT
KA780-QT
KA780-RT
KA780-ST
KA780-TT
KA780-UT
KA780-VT
KA780-WT
KA780-XT
KA780-YT
KA780-ZT
KA780-AU
KA780-BU
KA780-CU
KA780-DU
KA780-EU
KA780-FU
KA780-GU
KA780-HU
KA780-JU
KA780-KU
KA780-LU
KA780-MU
KA780-NU
KA780-PU
KA780-QU
KA780-RU
KA780-SU
KA780-TU
KA780-UU
KA780-VU
KA780-WU
KA780-XU
KA780-YU
KA780-ZU
KA780-AV
KA780-BV
KA780-CV
KA780-DV
KA780-EV
KA780-FV
KA780-GV
KA780-HV
KA780-JV
KA780-KV
KA780-LV
KA780-MV
KA780-NV
KA780-PV
KA780-QV
KA780-RV
KA780-SV
KA780-TV
KA780-UV
KA780-VV
KA780-WV
KA780-XV
KA780-YV
KA780-ZV
KA780-AW
KA780-BW
KA780-CW
KA780-DW
KA780-EW
KA780-FW
KA780-GW
KA780-HW
KA780-JW
KA780-KW
KA780-LW
KA780-MW
KA780-NW
KA780-PW
KA780-QW
KA780-RW
KA780-SW
KA780-TW
KA780-UW
KA780-VW
KA780-WW
KA780-XW
KA780-YW
KA780-ZW
KA780-AX
KA780-BX
KA780-CX
KA780-DX
KA780-EX
KA780-FX
KA780-GX
KA780-HX
KA780-JX
KA780-KX
KA780-LX
KA780-MX
KA780-NX
KA780-PX
KA780-QX
KA780-RX
KA780-SX
KA780-TX
KA780-UX
KA780-VX
KA780-WX
KA780-XX
KA780-YX
KA780-ZX
KA780-AY
KA780-BY
KA780-CY
KA780-DY
KA780-EY
KA780-FY
KA780-GY
KA780-HY
KA780-JY
KA780-KY
KA780-LY
KA780-MY
KA780-NY
KA780-PY
KA780-QY
KA780-RY
KA780-SY
KA780-TY
KA780-UY
KA780-VY
KA780-WY
KA780-XY
KA780-YY
KA780-ZY
KA780-AZ
KA780-BZ
KA780-CZ
KA780-DZ
KA780-EZ
KA780-FZ
KA780-GZ
KA780-HZ
KA780-JZ
KA780-KZ
KA780-LZ
KA780-MZ
KA780-NZ
KA780-PZ
KA780-QZ
KA780-RZ
KA780-SZ
KA780-TZ
KA780-UZ
KA780-VZ
KA780-WZ
KA780-XZ
KA780-YZ
KA780-ZZ
KA780-A0
KA780-B0
KA780-C0
KA780-D0
KA780-E0
KA780-F0
KA780-G0
KA780-H0
KA780-J0
KA780-K0
KA780-L0
KA780-M0
KA780-N0
KA780-P0
KA780-Q0
KA780-R0
KA780-S0
KA780-T0
KA780-U0
KA780-V0
KA780-W0
KA780-X0
KA780-Y0
KA780-Z0
KA780-A1
KA780-B1
KA780-C1
KA780-D1
KA780-E1
KA780-F1
KA780-G1
KA780-H1
KA780-J1
KA780-K1
KA780-L1
KA780-M1
KA780-N1
KA780-P1
KA780-Q1
KA780-R1
KA780-S1
KA780-T1
KA780-U1
KA780-V1
KA780-W1
KA780-X1
KA780-Y1
KA780-Z1
KA780-A2
KA780-B2
KA780-C2
KA780-D2
KA780-E2
KA780-F2
KA780-G2
KA780-H2
KA780-J2
KA780-K2
KA780-L2
KA780-M2
KA780-N2
KA780-P2
KA780-Q2
KA780-R2
KA780-S2
KA780-T2
KA780-U2
KA780-V2
KA780-W2
KA780-X2
KA780-Y2
KA780-Z2
KA780-A3
KA780-B3
KA780-C3
KA780-D3
KA780-E3
KA780-F3
KA780-G3
KA780-H3
KA780-J3
KA780-K3
KA780-L3
KA780-M3
KA780-N3
KA780-P3
KA780-Q3
KA780-R3
KA780-S3
KA780-T3
KA780-U3
KA780-V3
KA780-W3
KA780-X3
KA780-Y3
KA780-Z3
KA780-A4
KA780-B4
KA780-C4
KA780-D4
KA780-E4
KA780-F4
KA780-G4
KA780-H4
KA780-J4
KA780-K4
KA780-L4
KA780-M4
KA780-N4
KA780-P4
KA780-Q4
KA780-R4
KA780-S4
KA780-T4
KA780-U4
KA780-V4
KA780-W4
KA780-X4
KA780-Y4
KA780-Z4
KA780-A5
KA780-B5
KA780-C5
KA780-D5
KA780-E5
KA780-F5
KA780-G5
KA780-H5
KA780-J5
KA780-K5
KA780-L5
KA780-M5
KA780-N5
KA780-P5
KA780-Q5
KA780-R5
KA780-S5
KA780-T5
KA780-U5
KA780-V5
KA780-W5
KA780-X5
KA780-Y5
KA780-Z5
KA780-A6
KA780-B6
KA780-C6
KA780-D6
KA780-E6
KA780-F6
KA780-G6
KA780-H6
KA780-J6
KA780-K6
KA780-L6
KA780-M6
KA780-N6
KA780-P6
KA780-Q6
KA780-R6
KA780-S6
KA780-T6
KA780-U6
KA780-V6
KA780-W6
KA780-X6
KA780-Y6
KA780-Z6
KA780-A7
KA780-B7
KA780-C7
KA780-D7
KA780-E7
KA780-F7
KA780-G7
KA780-H7
KA780-J7
KA780-K7
KA780-L7
KA780-M7
KA780-N7
KA780-P7
KA780-Q7
KA780-R7
KA780-S7
KA780-T7
KA780-U7
KA780-V7
KA780-W7
KA780-X7
KA780-Y7
KA780-Z7
KA780-A8
KA780-B8
KA780-C8
KA780-D8
KA780-E8
KA780-F8
KA780-G8
KA780-H8
KA780-J8
KA780-K8
KA780-L8
KA780-M8
KA780-N8
KA780-P8
KA780-Q8
KA780-R8
KA780-S8
KA780-T8
KA780-U8
KA780-V8
KA780-W8
KA780-X8
KA780-Y8
KA780-Z8
KA780-A9
KA780-B9
KA780-C9
KA780-D9
KA780-E9
KA780-F9
KA780-G9
KA780-H9
KA780-J9
KA780-K9
KA780-L9
KA780-M9
KA780-N9
KA780-P9
KA780-Q9
KA780-R9
KA780-S9
KA780-T9
KA780-U9
KA780-V9
KA780-W9
KA780-X9
KA780-Y9
KA780-Z9
KA780-AA
KA780-BA
KA780-CA
KA780-DA
KA780-EA
KA780-FA
KA780-GA
KA780-HA
KA780-JA
KA780-KA
KA780-LA
KA780-MA
KA780-NA
KA780-PA
KA780-QA
KA780-RA
KA780-SA
KA780-TA
KA780-UA
KA780-VA
KA780-WA
KA780-XA
KA780-YA
KA780-ZA
KA780-AB
KA780-BB
KA780-CB
KA780-DB
KA780-EB
KA780-FB
KA780-GB
KA780-HB
KA780-JB
KA780-KB
KA780-LB
KA780-MB
KA780-NB
KA780-PB
KA780-QB
KA780-RB
KA780-SB
KA780-TB
KA780-UB
KA780-VB
KA780-WB
KA780-XB
KA780-YB
KA780-ZB
KA780-AC
KA780-BC
KA780-CC
KA780-DC
KA780-EC
KA780-FC
KA780-GC
KA780-HC
KA780-JC
KA780-KC
KA780-LC
KA780-MC
KA780-NC
KA780-PC
KA780-QC
KA780-RC
KA780-SC
KA780-TC
KA780-UC
KA780-VC
KA780-WC
KA780-XC
KA780-YC
KA780-ZC
KA780-AD
KA780-BD
KA780-CD
KA780-DD
KA780-ED
KA780-FD
KA780-GD
KA780-HD
KA780-JD
KA780-KD
KA780-LD
KA780-MD
KA780-ND
KA780-PD
KA780-QD
KA780-RD
KA780-SD
KA780-TD
KA780-UD
KA780-VD
KA780-WD
KA780-XD
KA780-YD
KA780-ZD
KA780-AE
KA780-BE
KA780-CE
KA780-DE
KA780-EE
KA780-FE
KA780-GE
KA780-HE
KA780-JE
KA780-KE
KA780-LE
KA780-ME
KA780-NE
KA780-PE
KA780-QE
KA780-RE
KA780-SE
KA780-TE
KA780-UE
KA780-VE
KA780-WE
KA780-XE
KA780-YE
KA780-ZE
KA780-AF
KA780-BF
KA780-CF
KA780-DF
KA780-EF
KA780-FF
KA780-GF
KA780-HF
KA780-JF
KA780-KF
KA780-LF
KA780-MF
KA780-NF
KA780-PF
KA780-QF
KA780-RF
KA780-SF
KA780-TF
KA780-UF
KA780-VF
KA780-WF
KA780-XF
KA780-YF
KA780-ZF
KA780-AG
KA780-BG
KA780-CG
KA780-DG
KA780-EG
KA780-FG
KA780-GG
KA780-HG
KA780-JG
KA780-KG
KA780-LG
KA780-MG
KA780-NG
KA780-PG
KA780-QG
KA780-RG
KA780-SG
KA780-TG
KA780-UG
KA780-VG
KA780-WG
KA780-XG
KA780-YG
KA780-ZG
KA780-AH
KA780-BH
KA780-CH
KA780-DH
KA780-EH
KA780-FH
KA780-GH
KA780-HH
KA780-JH
KA780-KH
KA780-LH
KA780-MH
KA780-NH
KA780-PH
KA780-QH
KA780-RH
KA780-SH
KA780-TH
KA780-UH
KA780-VH
KA780-WH
KA780-XH
KA780-YH
KA780-ZH
KA780-AJ
KA780-BJ
KA780-CJ
KA780-DJ
KA780-EJ
KA780-FJ
KA780-GJ
KA780-HJ
KA780-JJ
KA780-KJ
KA780-LJ
KA780-MJ
KA780-NJ
KA780-PJ
KA780-QJ
KA780-RJ
KA780-SJ
KA780-TJ
KA780-UJ
KA780-VJ
KA780-WJ
KA780-XJ
KA780-YJ
KA780-ZJ
KA780-AK
KA780-BK
KA780-CK
KA780-DK
KA780-EK
KA780-FK
KA780-GK
KA780-HK
KA780-JK
KA780-KK
KA780-LK
KA780-MK
KA780-NK
KA780-PK
KA780-QK
KA780-RK
KA780-SK
KA780-TK
KA780-UK
KA780-VK
KA780-WK
KA780-XK
KA780-YK
KA780-ZK
KA780-AL
KA780-BL
KA780-CL
KA780-DL
KA780-EL
KA780-FL
KA780-GL
KA780-HL
KA780-JL
KA780-KL
KA780-LL
KA780-ML
KA780-NL
KA780-PL
KA780-QL
KA780-RL
KA780-SL
KA780-TL
KA780-UL
KA780-VL
KA780-WL
KA780-XL
KA780-YL
KA780-ZL
KA780-AM
KA780-BM
KA780-CM
KA780-DM
KA780-EM
KA780-FM
KA780-GM
KA780-HM
KA780-JM
KA780-KM
KA780-LM
KA780-MM
KA780-NM
KA780-PM
KA780-QM
KA780-RM
KA780-SM
KA780-TM
KA780-UM
KA780-VM
KA780-WM
KA780-XM
KA780-YM
KA780-ZM
KA780-AN
KA780-BN
KA780-CN
KA780-DN
KA780-EN
KA780-FN
KA780-GN
KA780-HN
KA780-JN
KA780-KN
KA780-LN
KA780-MN
KA780-NN
KA780-PN
KA780-QN
KA780-RN
KA780-SN
KA780-TN
KA780-UN
KA780-VN
KA780-WN
KA780-XN
KA780-YN
KA780-ZN
KA780-AP
KA780-BP
KA780-CP
KA780-DP
KA780-EP
KA780-FP
KA780-GP
KA780-HP
KA780-JP
KA780-KP
KA780-LP
KA780-MP
KA780-NP
KA780-PP
KA780-QP
KA780-RP
KA780-SP
KA780-TP
KA780-UP
KA780-VP
KA780-WP
KA780-XP
KA780-YP
KA780-ZP
KA780-AQ
KA780-BQ
KA780-CQ
KA780-DQ
KA780-EQ
KA780-FQ
KA780-GQ
KA780-HQ
KA780-JQ
KA780-KQ
KA780-LQ
KA780-MQ
KA780-NQ
KA780-PQ
KA780-QQ
KA780-RQ
KA780-SQ
KA780-TQ
KA780-UQ
KA780-VQ
KA780-WQ
KA780-XQ
KA780-YQ
KA780-ZQ
KA780-AR
KA780-BR
KA780-CR
KA780-DR
KA780-ER
KA780-FR
KA780-GR
KA780-HR
KA780-JR
KA780-KR
KA780-LR
KA780-MR
KA780-NR
KA780-PR
KA780-QR
KA780-RR
KA780-SR
KA780-TR
KA780-UR
KA780-VR
KA780-WR
KA780-XR
KA780-YR
KA780-ZR
KA780-AS
KA780-BS
KA780-CS
KA780-DS
KA780-ES
KA780-FS
KA780-GS
KA780-HS
KA780-JS
KA780-KS
KA780-LS
KA780-MS
KA780-NS
KA780-PS
KA780-QS
KA780-RS
KA780-SS
KA780-TS
KA780-US
KA780-VS
KA780-WS
KA780-XS
KA780-YS
KA780-ZS
KA780-AT
KA780-BT
KA780-CT
KA780-DT
KA780-ET
KA780-FT
KA780-GT
KA780-HT
KA780-JT
KA780-KT
KA780-LT
KA780-MT
KA780-NT
KA780-PT
KA780-QT
KA780-RT
KA780-ST
KA780-TT
KA780-UT
KA780-VT
KA780-WT
KA780-XT
KA780-YT
KA780-ZT
KA780-AU
KA780-BU
KA780-CU
KA780-DU
KA780-EU
KA780-FU
KA780-GU
KA780-HU
KA780-JU
KA780-KU
KA780-LU
KA780-MU
KA780-NU
KA780-PU
KA780-QU
KA780-RU
KA780-SU
KA780-TU
KA780-UU
KA780-VU
KA780-WU
KA780-XU
KA780-YU
KA780-ZU
KA780-AV
KA780-BV
KA780-CV
KA780-DV
KA780-EV
KA780-FV
KA780-GV
KA780-HV
KA780-JV
KA780-KV
KA780-LV
KA780-MV
KA780-NV
KA780-PV
KA780-QV
KA780-RV
KA780-SV
KA780-TV
KA780-UV
KA780-VV
KA780-WV
KA780-XV
KA780-YV
KA780-ZV
KA780-AW
KA780-BW
KA780-CW
KA780-DW
KA780-EW
KA780-FW
KA780-GW
KA780-HW
KA780-JW
KA780-KW
KA780-LW
KA780-MW
KA780-NW
KA780-PW
KA780-QW
KA780-RW
KA780-SW
KA780-TW
KA780-UW
KA780-VW
KA780-WW
KA780-XW
KA780-YW
KA780-ZW
KA780-AX
KA780-BX
KA780-CX
KA780-DX
KA780-EX
KA780-FX
KA780-GX
KA780-HX
KA780-JX
KA780-KX
KA780-LX
KA780-MX
KA780-NX
KA780-PX
KA780-QX
KA780-RX
KA780-SX
KA780-TX
KA780-UX
KA780-VX
KA780-WX
KA780-XX
KA780-YX
KA780-ZX
KA780-AY
KA780-BY
KA780-CY
KA780-DY
KA780-EY
KA780-FY
KA780-GY
KA780-HY
KA780-JY
KA780-KY
KA780-LY
KA780-MY
KA780-NY
KA780-PY
KA780-QY
KA780-RY
KA780-SY
KA780-TY
KA780-UY
KA780-VY
KA780-WY
KA780-XY
KA780-YY
KA780-ZY
KA780-AZ
KA780-BZ
KA780-CZ
KA780-DZ
KA780-EZ
KA780-FZ
KA780-GZ
KA780-HZ
KA780-JZ
KA780-KZ
KA780-LZ
KA780-MZ
KA780-NZ
KA780-PZ
KA780-QZ
KA780-RZ
KA780-SZ
KA780-TZ
KA780-UZ
KA780-VZ
KA780-WZ
KA780-XZ
KA780-YZ
KA780-ZZ
KA780-A0
KA780-B0
KA780-C0
KA780-D0
KA780-E0
KA780-F0
KA780-G0
KA780-H0
KA780-J0
KA780-K0
KA780-L0
KA780-M0
KA780-N0
KA780-P0
KA780-Q0
KA780-R0
KA780-S0
KA780-T0
KA780-U0
KA780-V0
KA780-W0
KA780-X0
KA780-Y0
KA780-Z0
KA780-A1
KA780-B1
KA780-C1
KA780-D1
KA780-E1
KA780-F1
KA780-G1
KA780-H1
KA780-J1
KA780-K1
KA780-L1
KA780-M1
KA780-N1
KA780-P1
KA780-Q1
KA780-R1
KA780-S1
KA780-T1
KA780-U1
KA780-V1
KA780-W1
KA780-X1
KA780-Y1
KA780-Z1
KA780-A2
KA780-B2
KA780-C2
KA780-D2
KA780-E2
KA780-F2
KA780-G2
KA780-H2
KA780-J2
KA780-K2
KA780-L2
KA780-M2
KA780-N2
KA780-P2
KA780-Q2
KA780-R2
KA780-S2
KA780-T2
KA780-U2
KA780-V2
KA780-W2
KA780-X2
KA780-Y2
KA780-Z2
KA780-A3
KA780-B3
KA780-C3
KA780-D3
KA780-E3
KA780-F3
KA780-G3
KA780-H3
KA780-J3
KA780-K3
KA780-L3
KA780-M3
KA780-N3
KA780-P3
KA780-Q3
KA780-R3
KA780-S3
KA780-T3
KA780-U3
KA780-V3
KA780-W3
KA780-X3
KA780-Y3
KA780-Z3
KA780-A4
KA780-B4
KA780-C4
KA780-D4
KA780-E4
KA780-F4
KA780-G4
KA780-H4
KA780-J4
KA780-K4
KA780-L4
KA780-M4
KA780-N4
KA780-P4
KA780-Q4
KA780-R4
KA780-S4
KA780-T4
KA780-U4
KA780-V4
KA780-W4
KA780-X4
KA780-Y4
KA780-Z4
KA780-A5
KA780-B5
KA780-C5
KA780-D5
KA780-E5
KA780-F5
KA780-G5
KA780-H5
KA780-J5
KA780-K5
KA780-L5
KA780-M5
KA780-N5
KA780-P5
KA780-Q5
KA780-R5
KA780-S5
KA780-T5
KA780-U5
KA780-V5
KA780-W5
KA780-X5
KA780-Y5
KA780-Z5
KA780-A6
KA780-B6
KA780-C6
KA780-D6
KA780-E6
KA780-F6
KA780-G6
KA780-H6
KA780-J6
KA780-K6
KA780-L6
KA780-M6
KA780-N6
KA780-P6
KA780-Q6
KA780-R6
KA780-S6
KA780-T6
KA780-U6
KA780-V6
KA780-W6
KA780-X6
KA780-Y6
KA780-Z6
KA780-A7
KA780-B7
KA780-C7
KA780-D7
KA780-E7
KA780-F7
KA780-G7
KA780-H7
KA780-J7
KA780-K7
KA780-L7
KA780-M7
KA780-N7
KA780-P7
KA780-Q7
KA780-R7
KA780-S7
KA780-T7
KA780-U7
KA780-V7
KA780-W7
KA780-X7
KA780-Y7
KA780-Z7
KA780-A8
KA780-B8
KA780-C8
KA780-D8
KA780-E8
KA780-F8
KA780-G8
KA780-H8
KA780-J8
KA780-K8
KA780-L8
KA780-M8
KA780-N8
KA780-P8
KA780-Q8
KA780-R8
KA780-S8
KA780-T8
KA780-U8
KA780-V8
KA780-W8
KA780-X8
KA780-Y8
KA780-Z8
KA780-A9
KA780-B9
KA780-C9
KA780-D9
KA780-E9
KA780-F9
KA780-G9
KA780-H9
KA780-J9
KA780-K9
KA780-L9
KA780-M9
KA780-N9
KA780-P9
KA780-Q9
KA780-R9
KA780-S9
KA780-T9
KA780-U9
KA780-V9
KA780-W9
KA780-X9
KA780-Y9
KA780-Z9
KA780-AA
KA780-BA
KA780-CA
KA780-DA
KA780-EA
KA780-FA
KA780-GA
KA780-HA
KA780-JA
KA780-KA
KA780-LA
KA780-MA
KA780-NA
KA780-PA
KA780-QA
KA780-RA
KA780-SA
KA780-TA
KA780-UA
KA780-VA
KA780-WA
KA780-XA
KA780-YA
KA780-ZA
KA780-AB
KA780-BB
KA780-CB
KA780-DB
KA780-EB
KA780-FB
KA780-GB
KA780-HB
KA780-JB
KA780-KB
KA780-LB
KA780-MB
KA780-NB
KA780-PB
KA780-QB
KA780-RB
KA780-SB
KA780-TB
KA780-UB
KA780-VB
KA780-WB
KA780-XB
KA780-YB
KA780-ZB
KA780-AC
KA780-BC
KA780-CC
KA780-DC
KA780-EC
KA780-FC
KA780-GC
KA780-HC
KA780-JC
KA780-KC
KA780-LC
KA780-MC
KA780-NC
KA780-PC
KA780-QC
KA780-RC
KA780-SC
KA780-TC
KA780-UC
KA780-VC
KA780-WC
KA780-XC
KA780-YC
KA780-ZC
KA780-AD
KA780-BD
KA780-CD
KA780-DD
KA780-ED
KA780-FD
KA780-GD
KA780-HD
KA780-JD
KA780-KD
KA780-LD
KA780-MD
KA780-ND
KA780-PD
KA780-QD
KA780-RD
KA780-SD
KA780-TD
KA780-UD
KA780-VD
KA780-WD
KA780-XD
KA780-YD
KA780-ZD
KA780-AE
KA780-BE
KA780-CE
KA780-DE
KA780-EE
KA780-FE
KA780-GE
KA780-HE
KA780-JE
KA780-KE
KA780-LE
KA780-ME
KA780-NE
KA780-PE
KA780-QE
KA780-RE
KA780-SE
KA780-TE
KA780-UE
KA780-VE
KA780-WE
KA780-XE
KA780-YE
KA780-ZE
KA780-AF
KA780-BF
KA780-CF
KA780-DF
KA780-EF
KA780-FF
KA780-GF
KA780-HF
KA780-JF
KA780-KF
KA780-LF
KA780-MF
KA780-NF
KA780-PF
KA780-QF
KA780-RF
KA780-SF
KA780-TF
KA780-UF
KA780-VF
KA780-WF
KA780-XF
KA780-YF
KA780-ZF
KA780-AG
KA780-BG
KA780-CG
KA780-DG
KA780-EG
KA780-FG
KA780-GG
KA780-HG
KA780-JG
KA780-KG
KA780-LG
KA780-MG
KA780-NG
KA780-PG
KA780-QG
KA780-RG
KA780-SG
KA780-TG
KA780-UG
KA780-VG
KA780-WG
KA780-XG
KA780-YG
KA780-ZG
KA780-AH
KA780-BH
KA780-CH
KA780-DH
KA780-EH
KA780-FH
KA780-GH
KA780-HH
KA780-JH
KA780-KH
KA780-LH
KA780-MH
KA780-NH
KA780-PH
KA780-QH
KA780-RH
KA780-SH
KA780-TH
KA780-UH
KA780-VH
KA780-WH
KA780-XH
KA780-YH
KA780-ZH
KA780-AJ
KA780-BJ
KA780-CJ
KA780-DJ
KA780-EJ
KA780-FJ
KA780-GJ
KA780-HJ
KA780-JJ
KA780-KJ
KA780-LJ
KA780-MJ
KA780-NJ
KA780-PJ
KA780-QJ
KA780-RJ
KA780-SJ
KA780-TJ
KA780-UJ
KA780-VJ
KA780-WJ
KA780-XJ
KA780-YJ
KA780-ZJ
KA780-AK
KA780-BK
KA780-CK
KA780-DK
KA780-EK
KA780-FK
KA780-GK
KA780-HK
KA780-JK
KA780-KK
KA780-LK
KA780-MK
KA780-NK
KA780-PK
KA780-QK
KA780-RK
KA780-SK
KA780-TK
KA780-UK
KA780-VK
KA780-WK
KA780-XK
KA780-YK
KA780-ZK
KA780-AL
KA780-BL
KA780-CL
KA780-DL
KA780-EL
KA780-FL
KA780-GL
KA780-HL
KA780-JL
KA780-KL
KA780-LL
KA780-ML
KA780-NL
KA780-PL
KA780-QL
KA780-RL
KA780-SL
KA780-TL
KA780-UL
KA780-VL
KA780-WL
KA780-XL
KA780-YL
KA780-ZL
KA780-AM
KA780-BM
KA780-CM
KA780-DM
KA780-EM
KA780-FM
KA780-GM
KA780-HM
KA780-JM
KA780-KM
KA780-LM
KA780-MM
KA780-NM
KA780-PM
KA780-QM
KA780-RM
KA780-SM
KA780-TM
KA780-UM
KA780-VM
KA780-WM
KA780-XM
KA780-YM
KA780-ZM
KA780-AN
KA780-BN
KA780-CN
KA780-DN
KA780-EN
KA780-FN
KA780-GN
KA780-HN
KA780-JN
KA780-KN
KA780-LN
KA780-MN
KA780-NN
KA780-PN
KA780-QN
KA780-RN
KA780-SN
KA780-TN
KA780-UN
KA780-VN
KA780-WN
KA780-XN
KA780-YN
KA780-ZN
KA780-AP
KA780-BP
KA780-CP
KA780-DP
KA780-EP
KA780-FP
KA780-GP
KA780-HP
KA780-JP
KA780-KP
KA780-LP
KA780-MP
KA780-NP
KA780-PP
KA780-QP
KA780-RP
KA780-SP
KA780-TP
KA780-UP
KA780-VP
KA780-WP
KA780-XP
KA780-YP
KA780-ZP
KA780-AQ
KA780-BQ
KA780-CQ
KA780-DQ
KA780-EQ
KA780-FQ
KA780-GQ
KA780-HQ
KA780-JQ
KA780-KQ
KA780-LQ
KA780-MQ
KA780-NQ
KA780-PQ
KA780-QQ
KA780-RQ
KA780-SQ
KA780-TQ
KA780-UQ
KA780-VQ
KA780-WQ
KA780-XQ
KA780-YQ
KA780-ZQ
KA780-AR
KA780-BR
KA780-CR
KA780-DR
KA780-ER
KA780-FR
KA780-GR
KA780-HR
KA780-JR
KA780-KR
KA780-LR
KA780-MR
KA780-NR
KA780-PR
KA780-QR
KA780-RR
KA780-SR
KA780-TR
KA780-UR
KA780-VR
KA780-WR
KA780-XR
KA780-YR
KA780-ZR
KA780-AS
KA780-BS
KA780-CS
KA780-DS
KA780-ES
KA780-FS
KA780-GS
KA780-HS
KA780-JS
KA780-KS
KA780-LS
KA780-MS
KA780-NS
KA780-PS
KA780-QS
KA780-RS
KA780-SS
KA780-TS
KA780-US
KA780-VS
KA780-WS
KA780-XS
KA780-YS
KA780-ZS
KA780-AT
KA780-BT
KA780-CT
KA780-DT
KA780-ET
KA780-FT
KA780-GT
KA780-HT
KA780-JT
KA780-KT
KA780-LT
KA780-MT
KA780-NT
KA780-PT
KA780-QT
KA780-RT
KA780-ST
KA780-TT
KA780-UT
KA780-VT
KA780-WT
KA780-XT
KA780-YT
KA780-ZT
KA780-AU
KA780-BU
KA780-CU
KA780-DU
KA780-EU
KA780-FU
KA780-GU
KA780-HU
KA780-JU
KA780-KU
KA780-LU
KA780-MU
KA780-NU
KA780-PU
KA780-QU
KA780-RU
KA780-SU
KA780-TU
KA780-UU
KA780-VU
KA780-WU
KA780-XU
KA780-YU
KA780-ZU
KA780-AV
KA780-BV
KA780-CV
KA780-DV
KA780-EV
KA780-FV
KA780-GV
KA780-HV
KA780-JV
KA780-KV
KA780-LV
KA780-MV
KA780-NV
KA780-PV
KA780-QV
KA780-RV
KA780-SV
KA780-TV
KA780-UV
KA780-VV
KA780-WV
KA780-XV
KA780-YV
KA780-ZV
KA780-AW
KA780-BW
KA780-CW
KA780-DW
KA780-EW
KA780-FW
KA780-GW
KA780-HW
KA780-JW
KA780-KW
KA780-LW
KA780-MW
KA780-NW
KA780-PW
KA780-QW
KA780-RW
KA780-SW
KA780-TW
KA780-UW
KA780-VW
KA780-WW
KA780-XW
KA780-YW
KA780-ZW
KA780-AX
KA780-BX
KA780-CX
KA780-DX
KA780-EX
KA780-FX
KA780-GX
KA780-HX
KA780-JX
KA780-KX
KA780-LX
KA780-MX
KA780-NX
KA780-PX
KA780-QX
KA780-RX
KA780-SX
KA780-TX
KA780-UX
KA780-VX
KA780-WX
KA780-XX
KA780-YX
KA780-ZX
KA780-AY
KA780-BY
KA780-CY
KA780-DY
KA780-EY
KA780-FY
KA780-GY
KA780-HY
KA780-JY
KA780-KY
KA780-LY
KA780-MY
KA780-NY
KA780-PY
KA780-QY
KA780-RY
KA780-SY
KA780-TY
KA780-UY
KA780-VY
KA780-WY
KA780-XY
KA780-YY
KA780-ZY
KA780-AZ
KA780-BZ
KA780-CZ
KA780-DZ
KA780-EZ
KA780-FZ
KA780-GZ
KA780-HZ
KA780-JZ
KA780-KZ
KA780-LZ
KA780-MZ
KA780-NZ
KA780-PZ
KA780-QZ
KA780-RZ
KA780-SZ
KA780-TZ
KA780-UZ
KA780-VZ
KA780-WZ
KA780-XZ
KA780-YZ
KA780-ZZ

If memory modules of unlike sizes are installed in the same system, install the largest capacity modules first. If a combination of four 64 Mb and 16 Mb modules have already been installed, one 4 Mb module may be installed in slot 8.

Use of L9200 Memory Load Modules

The L9200 is a memory array load module. It is used to put the minimum specified load on the appropriate regulator. This module is to be used in systems that have an insufficient number of array modules for correct loading. If there is no memory array module in slot 5 or 6, then an L9200 is required in those slots. There is an exception to this rule for systems that are using M.0% version 2 and 3 packs. For these cases, a memory array module must be installed in slot 6, and if slot 7 is empty, it must have an L9200.

7.5.2.3 4-slot Backplane - The KA64, which performs critical timing and data transfers from the KA7 to the CPU, is capable of supporting two DSSI AA SCSI Adapters (KFBTA). The first unit (KFBTA0) is the SBI adapter for the I/O backplane. The second unit (KFBTA1) is used for optional VAX expansion capabilities.

7.5.2.4 I/O Backplane - The I/O backplane is a dedicated, pre-wired SBI, UNIBUS backplane that can support one CI750-MA, one DW780-MA, one DW780-MB, and a BDUB-AA.

1. The CI750-MA is used for connection to a clusters environment. The CI cables connect to the rear bulkhead of the CPU cabinet.

2. The DW780-MA is included with four kernel systems. The UNIBUS for this is connected to the BA11 UNIBUS drawer in the front end cabinet. This adapter is dedicated for front end use only.

3. The DW780-MB is designed for UNIBUS expansion out of the I/O backplane. The UNIBUS for this DW connects to a BA11 drawer in a UNIBUS expansion cabinet mounted to the left of the front end cabinet.

4. The BDUB-AA is comprised of a DW780-MA and a H7A50-A. These options result in a reserved area of the I/O backplane which supports a TU81 controller module. Note that these three SBI UNIBUS slots are wired specifically for these options (UDA50 and TU81 controller) and will not work with other UNIBUS options. Furthermore, this UNIBUS is terminated within the I/O backplane (M9302 in slot 11 and is not intended to be expanded beyond the CPU cabinet on a permanent basis.

   Contact your local field support organization for information on expanding this UNIBUS for troubleshooting purposes.

5. The M9040 module installed in slot 5 provides termination for the SBI. This module must be removed if Slot is expanded to an expansion cabinet.

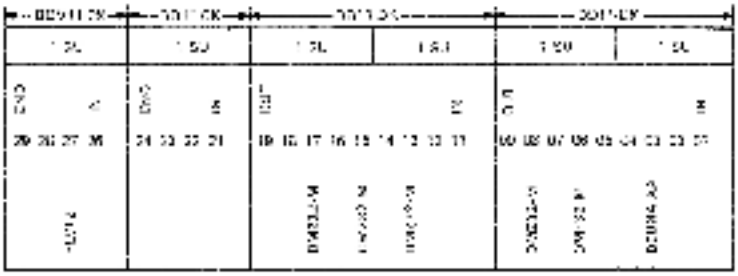


Figure 7-5 BA11-AF Front End Cabinet

7.5.3 Front End Cabinet

The Front End Cabinet (FEC) contains the console RL02 disk, the RLV12 controller, BA11-AF expansion boxes and UNIBUS backplanes. Within the FEC there is space reserved for mounting a 50Hz transformer.

The BA11-AF is controlled by the DW730-BA (DW0) in the CPU cabinet and is configured with five UNIBUS system units and one QBus system unit for the console. The standard kernel system shown above, includes:

1. DZ11A (1)
2. LPV11 (1)
3. LPV11 (4)

This is the MAXIMUM configuration allowed within the BA11-AL mounting box and is not expandable. You can, however, alter the UNIBUS configuration within the BA11-AL by replacing some of the options. Furthermore, this UNIBUS can be expanded to a second BA11-A drawer mounted in a UNIBUS expansion cabinet. Refer to 'BA11-AL/AM EXPANSION RULES' for more details on configuration options within the Front End BA11-AL.

7.5.4 BA11-AL/AM EXPANSION RULES

The BA11-AL/AM can accommodate six system units, DD11-DK or DD11-DK. The maximum output power in the BA11-A box cannot exceed 500 watts. The power supply in the BA11-A box is partitioned into two areas; the first area containing the first four system units and the second area containing the remaining two system units. The power supply is rated at 15 V 95 A. The systems units are as follows:

System unit 1-2	+5 V	42 A
System unit 3-4	+5 V	22 A
	-12 V	1 A
System unit 1-4	-15 V	1 A
System unit 5-6	+5 V	28 A
	+15 V	5 A
	+15 V	1.5 A

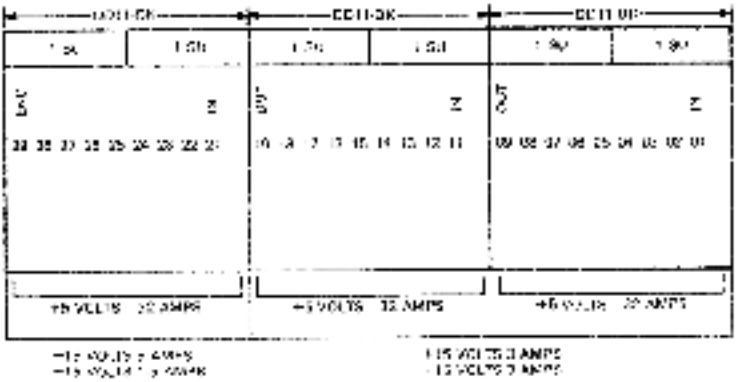


Figure 7-6 DBII-BI/AN Backplane Diagram

Configuration Rules for DBII-BI/AN

When adding an option or backplane to a DBII-A expansion box the following rules apply:

1. In addition to the regular configuring rules on the Asperega available for the backplane, the DBII-A expansion box requires that the power drawn by options be known and used as a configuring requirement. The total power drawn by UNIBUS options mounted in the expansion box cannot exceed 500 watts (see discussion below).

2. Quare hex-sized options in the backplane starting in slot 3. Slots 1, 2, and 9 are available only for use by quad-size options, so when configuring quad-sized options in these slots violates rule 3.

3. Any single module option that produces more than 40 watts requires that the slot adjacent to the component side of the option be unoccupied (due to cooling restrictions). Dual-module options (i.e., DEUNA) that produce over 40 watts can be mounted in consecutive slots, and require the slot adjacent to the component side of the dual-module option be unoccupied.

4. Standard UNIBUS configuration rules apply to any UNIBUS on the system. See the 'PDP 11 BUS HANDBOOK' or the Field Service 'MAINDEC' program for assistance.

Example 7-1: Power consumption for the DMP32

The DMP32 Communications Controller draws 6 Amps @ +5 V, 0.25 Amps @ +15 V, and .50 Amps @ -15 V. Using these figures as input to the formula the following results are achieved:

```
6 X 5   = 30 (watts)
.5 X 15 = 7.5 (watts)
.5 X 15 = 7.5 (watts)
-------------------
Option total = 55 watts
```

This result is important in two ways. One, because the DWR32 option produces more than 40 watts, the previous slot (slot adjacent to the component side of the module) needs to be unoccupied. Second, the power consumed by the DWR32 (5I) is subtracted from the available total of 500 watts. Therefore, to add the DWR32 to an unpopulated DD11-DK backplane in a BA11-A expansion box, the option will require two slots; slot 2 for the 40 watt rule, slot 3 for the module. This leaves 445 watts remaining. Adding a second DWR32 to the backplane would require two slots, slot 4 remains unoccupied and the second DWR32 module is configured into slot 5.

Proceed in the same manner for additional UNIBUS options, tracking the total power consumption for the box, and following the slot rule based on each options power requirements.

7.5.3 UNIBUS Expansion

There are three ways to utilize the UNIBUS expansion capabilities of the system.

1. Use the DW780-KB installed in the I/O backplane.

   The DW780-KB mounts in the I/O backplane (DW). Order the H9652-F UNIBUS expansion cabinet and DD11-DK backplanes. These cabinets will mount to the left side of the front end cabinet. See diagram of UNIBUS expansion configuration.

2. To use the DW780-AA/AH (installed in SBI expansion cabinet).

   This variation mounts in an H9652-C series SBI expansion cabinet. Order H9652-F series UNIBUS expansion cabinets and DD11-DK backplanes. The UNIBUS cabinets will mount to the right of the base H9652-C series SBI cabinet.

3. To expand the front-end UNIBUS (DW0).

   The UNIBUS in the front end BA11-A drawer (DW0) may be expanded to a second BA11-A drawer in a UNIBUS expansion cabinet. Order H9652-F Series UNIBUS expansion cabinets and DD11-DK backplanes. The UNIBUS cabinets will mount to the left of the front end cabinet (see diagram below).

   The standard UNIBUS configuration rules apply (i.e., bus loads, bus length). See the section 'BA11-AL/AK EXPANSION BOXES' for more information on configuring the BA11-A UNIBUS expansion box. Also, the Field Service 'PAUSE' program may be used for reference.

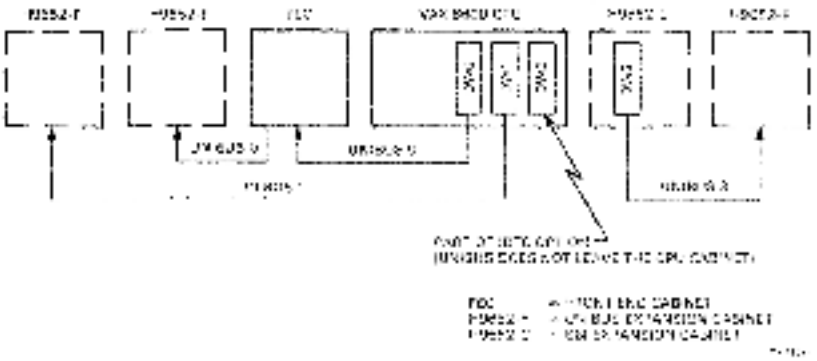


Figure 7-7 CNIBUS Expansion Configurations

7.5.6 SBI Expansion

The system is capable of supporting two independent SBI adapters. There are two ways to obtain these SBI expansion capabilities.

1. Expand the SBI adapter included in the CPU.

   a. The configuration rule for the first SBI allows a maximum of one H9652-C SBI expansion cabinet (4 option panel spaces) which mounts to the right of the CPU cabinet.

   b. Refer to system SBI configuration rules for more details.

2. Order the optional SBI adapter, DW86-AA.

   a. The configuration rule for the second SBI allows a maximum of two H9652-C SBI expansion cabinets (8 SBI option slots) which mount to the right of the CPU cabinet.

   b. Refer to system SBI configuration rules for more details.

7.5.5.1 SBI Configuration Rules -

1. Only one SBI can be expanded per system.

2. Maximum of two CI780s per system (1 each on the internal SBI or the optional SBI). See cluster rules for more details.

3. Maximum of two CI780s per SBI (only one if a DW780 is also connected).

4. Maximum of four DW780s per SBI (only one if a CI780 is also connected).

5. Maximum of four RB780As per SBI.

6. Maximum of four DW780s per SBI.

7. Must extending the internal SBI (SBIA-0) from the CPU cabinet to the 8965-C SBI expansion cabinet, be sure to remove the SBI terminator (M9048) from slot 5 of the I/O backplane.

7.5.6.2 SBI IR Level Assignments - The chart below is suggested as a guideline for assigning IR levels on SBIs. On a system with two separate SBIs, configure each SBI independently.

Absolute IR levels and unusual IR levels do not matter. Relative IR levels determine relative priority and do not matter. High number IR levels mean low priority in competing for SBI access. IR 0 is the highest priority and IR 16 is the lowest priority.

The following restrictions apply:

1. IR 00 is reserved for HALT

2. IR 01 is reserved for DMA return (accepted by SBIA)

3. IR 02 is reserved for the CPU (actually SBIA) (see note 2)

VAX 8600 SYSTEM SBI IR LEVEL ASSIGNMENT CHART

IR level	SBI Device
0	HALT
1	DMA return (SBIA)
2	CPU (SBIA) (see note 2)
3	first CBA (DW780)
4	second CBA
5	third CBA
6	fourth CBA
7	- - -
8	first MBA (for disks)
9	second MBA (for tapes)
10	third MBA
11	fourth MBA
12	- - -
13	first CI780
14	first DW780 (or second CI780)
15	second DW780
16	- - -

NOTES

1. Refer to section SBI NEXUS and INTERNAL OPTIONS for information on setting the TR level for a particular option. This section will include information on jumper settings.

2. Where the CPU(SBIA) TR level is 2, the SBI ID number remains at 16.

3. The DW780 and CI780 TR levels may be interchanged, however, if a DW780 and a CI780 exist on the same SBI then the CI780 must be at the higher priority (as shown).

7.5.7 Cluster Rules

1. Every CPU in a cluster must be able to communicate with every other CPU. (i.e. CPU must have a path via CI connected to a common star coupler.)

2. A CPU may be a member of only one cluster at a time.

3. HSC50s connected to a CPU on a star coupler other than the star couplers on the public cluster are considered 'private HSC50s'.

4. Disks connected to private HSC50s cannot be made cluster accessible.

5. VMS can presently support only two CIs per CPU. Three or more are possible in a future release.

6. Only 16 nodes per star coupler.

7. Each node connected to a star coupler must have a unique node number and node name.

7.6 PM PROCEDURES

The following PM procedure has been developed in an effort to increase the MTBF for the VAX 8600/8650 systems. This procedure should be followed in the field on a consistent basis to ensure maximum reliability.

It is recommended that error logs be checked and diagnostics be run prior to performing this procedure to ensure that no problems are injected into the system during this procedure.

7.6.1 Summary of Quarterly PM Procedures

1. Clean air vents, check fans and check voltages in BA boxes.
2. Check system cables for damage.
3. Clean and replace filters.
4. Verify revisions.
5. Check system power.
6. Perform scheduled device PMs.
7. Run system exercisers.

7.6.2 Summary of Annual PM Procedures

1. Run margins.
2. Run system diagnostics.
3. Replace RL02 filter absolute filter

7.6.3 Quarterly PM Procedures

1. Clean air vents, check fans and check voltages in BA boxes.
   a. Visually inspect air vents and clean as necessary.
   b. Ensure fans are operating correctly and replace as needed.

c. Measure voltages in the BA1x box using a digital voltmeter (DVM). The table below lists the color code, and associated backplane pin, used on each of the power harnesses within the BA11 expansion drawer.

Color	Voltage	Backplane Pin
BLACK	GROUND	C2
RED	- 5 VOLTS	B2
GRAY	-15 VOLTS	C01
WHITE	-15 VOLTS	C01
BLUE	-15 VOLTS	C02
GREEN	-15 VOLTS	C02
BROWN	- 5 VOLTS	C01
VIOLET	AC LOW	CN1
YELLOW	AC LOW	LV1

NOTE

Log and retain these power measurements for comparison at each PM performance, for indications of power degradation. The BT140 is an FRU as a complete unit and voltage adjustments are to be done at the factory only.

2. Check system cables for damage.

- Visually inspect system cables for fraying, crimping, or any other types of damage. Replace as necessary.

3. Clean and replace filters.

a. CPU card cage filter - Power down the system. Turn the two black catches at the bottom of the card cage support assembly so they are parallel to the support assembly. Slide the filter out of the CPU cabinet. Use a vacuum to clean the filter, then reinstall the filter.

b. Front and cabinet - Open front and rear doors of the front end cabinet. Remove the filters mounted on the inside of the doors by pulling them away from the velcro strips holding them in place. Insert the new filters, part numbers 12-11294-01 (1), 12-11295-01 (3), and 12-11295-02 (1).

c. RF03 ventilator - Remove the 6 screws holding the front panel in place. Remove the front bezel. Replace the ventilator located on the right front side of the drive (PN 74-15297-00).

d. CPU Air Mufflers - Insure that free air flow exists in the air mufflers mounted on the rear doors of the CPU.

4. Verify revisions.

   a. Check system revisions by referencing the Revision Matrix documents in the microfiche, EK-PM-8650-C-0 for the VAX 8650 or EK-PM-8600-C-0 for the VAX 8600.

   b. Install any mandatory ECOs.

   c. Insure that the latest XL02 console release is being used on the system.

5. Check system power.

   - Measure system regulator power and ground current by use of the env power command (>>>SHOW POWER). Retain this for comparison at each quarterly PM, for indications of possible system power degradation.

6. Perform scheduled device PMs.

   - Refer to microfiche PM listings.

7. Run system exerciser (EVKAA).

   - Run the VAX 8XXX system exerciser diagnostic, EVKAA, for 1 pass, referring to the diagnostic microfiche listing for specific operating instructions.

```
>>>@FLSAA
DS> ADT KA66 HUB KAC Y Y 0 1
DS> TOC KAC
DS> @ EJKAA/PASS=1
```

7.6.4 Annual PM Procedures

1. Replace RL02 absolute filter

   - Locate the 2 plenum springs and plenum cover latches. Lift one of the plenum latches. While holding the plenum spring, squeeze the latch and remove the latch from the bracket. Remove the plenum spring then repeat the procedure for the 2nd latch and spring. The plenum cover may now be removed by pulling the cover forward. Use a screwdriver to pry the left side of the filter out slightly. The filter can then be removed by rocking it from side to side until it slides free. Replace the filter (PN 12-21097-03).

2. Run system diagnostics.

   - Executes 1 pass of the TSTCPU.COM file. This will take about 45 minutes and test the suspected CPU.

```
>>>@ISTCPU
```

3. Run margins.

   - Run one pass of the ISTCPU script with voltage margins high. Repeat with voltage margins low.

```
>>>DIAG
DC>SET MARG HI
DC>@TSTCPU
DC>SET MARG NO
DC>@TSTCPU

DC>SET MARG LO
```

4. Execute CPU MACRO diagnostics.

   a. Load the Diagnostic Supervisor and ATTACH the CPU.

```
>>>@EDSAA
DS> ATT DADG (RUN RAD Y Y 0 )
DS> SEL RAD
```

   b. Run 1 pass of the CPU macro diagnostics.

```
DS> R diagnostic_name
      Diagnostics to be run are:
      EVKAB
      EVKAC
      EVKAD
      EVKAE
      EUKAX
```

7.6.5 VAX 6400/6650 PM Checklist

TASK	QUARTERLY			
BA 80X35				
1. Air vents cleared				
2. Fans operating correctly				
3. Voltages checked				
KERNEL				
1. System cables inspected				
2. CPU cold cage filter cleaned				
FRONT END				
1. Door filters replaced				
2. RF02 prefilter replaced				
SYSTEM				
1. Revisions verified per EK				
2. System power checked				
DEVICES				
1. All system PMs performed				
TESTING				
1. 3DMAX runs error free				

	ANNUAL
POWER END	
1. RL02 absolute filter replaced	
TESTING	
1. High margins performed	
2. Low margins performed	
3. VSCCPU runs at normal margin	
4. MSCB0 diags run error free	

CHAPTER 8

SYSTEM CLOCKS

8.1 CLOCK INTRODUCTION

Two types of clock modules are used in the VAX 8600/8650 processors, the L0217 and L0231. There are two versions of the L0217 module, revision C, which will be phased out over time, and revision E. The VAX 8600 can use either revision of the L0217, or the L0231, but with the L0231, use is restricted to the lower frequencies. The VAX 8650 can only use the L0231.

The main difference between the two L0217 versions is the source of the clock signal. The revision C clock is provided by a 50 MHz oscillator, a phase lock loop, or an external clock. The phase lock loop provides clocking rates between 40 and 64 MHz. The revision E L0217 has no phase locked loop, but has four oscillators and an external clock.

The L0231 module is similar to the L0217 rev. E, but has six oscillators instead of four, and the delays are different, being adjusted to the faster clocking rate of the VAX 8650. The L0231 also has an external clock input.

8.2 CLOCK CONSOLE COMMANDS

The clock is controlled by the following console commands:

- SET CLOCK
- SET SCSI
- SHOW CLOCK
- START/STOP CPU
- START/STOP SYSTEM

These commands are part of the General Command Set, and may be found in Chapter 10, Console Software and Commands.

The SET SCSI command works in conjunction with the DEC/STD/MAX console command to allow stepping on a microcycle. See the DEC command set (Chapter 10) for DSSC217/MASM.

8.3 CLOCK OUTPUTS

Table 8-1 lists the VAX 8600 clock and Table 8-2 lists the clock power signals. For the remainder of the clock outputs, see VAX 8600/8650 System Clocks Technical Description, EK-SASYS-TD, Appendix A.

Table 8-1 WBus Enable Signals

Signal Name	Source Module	Slot	Pin	Destination Module	B/P	Slot	Pin
CLK F04 WSL0 ENABLE L	CLK	11	B38	FEA	02	08	B92
CLK B6S WSL8 ENABLE L	CLK	11	D36	KDX	02	09	A45
CLK E0F WSL9 ENABLE L	CLK	11	B37	BDF	03	10	C10
CLK 1C0 WSL2 ENABLE L	CLK	11	B34	JDP	02	14	A91

Table 8-2 Clock Reset Signals

Signal Name	Source Module	Slot	Pin	Destination Module	B/P	Slot	Pin
CLK RESET 141 A	CLK	11	A85	MJJ	02	16	B50
CLK RESET 141 A L	CLK	11	A85	MAP	02	17	D50
CLK RESET 141 A L	CLK	11	A85	MCC	02	18	B50
CLK RESET 141 B L	CLK	11	A86	JCJ	02	12	B50
CLK RESET 141 B L	CLK	11	A86	JLA	02	13	B50
CLK RESET 141 B L	CLK	11	A86	JDP	02	14	B50
CLK RESET 141 B L	CLK	11	A86	JHC	02	15	B50
CLK RESET 141 C L	CLK	11	A87	KBJ	02	06	B50
CLK RESET 141 C L	CLK	11	A87	BBE	02	08	D50
CLK RESET 141 C L	CLK	11	A87	KDP	02	10	B50
CLK RESET 141 D L	CLK	11	A88	CHA	03	03	B50
CLK RESET 141 D L	CLK	11	A88	CEB	02	04	D50
CLK RESET 141 D L	CLK	11	A88	KBC	02	05	B50
CLK RESET 141 E L	CLK	11	A89	FLM	02	07	B50
CLK RESET 141 E L	CLK	11	A89	FEA	02	08	D50
CLK SDA0 RESET 141 L	CLK	11	A91	SBA	03	01	C50
CLK SDA1 RESET 141 L	CLK	11	A93	SBA	03	05	C50
CLK SDA2 RESET 141 L	CLK	11	B14	-	-	-	-
CLK SDA3 RESET 141 L	CLK	11	B15	-	-	-	-

8.4 CLOCK BLOCK DIAGRAMS

Block diagrams are included for the clock distribution system, the LU217 Rev. C5 module, the LU221/LU217 Rev. E modules, backplane clock distribution, and clock control logic, the CIC ASA.

NOTE

The CIC ASA is the same for all of the modules.

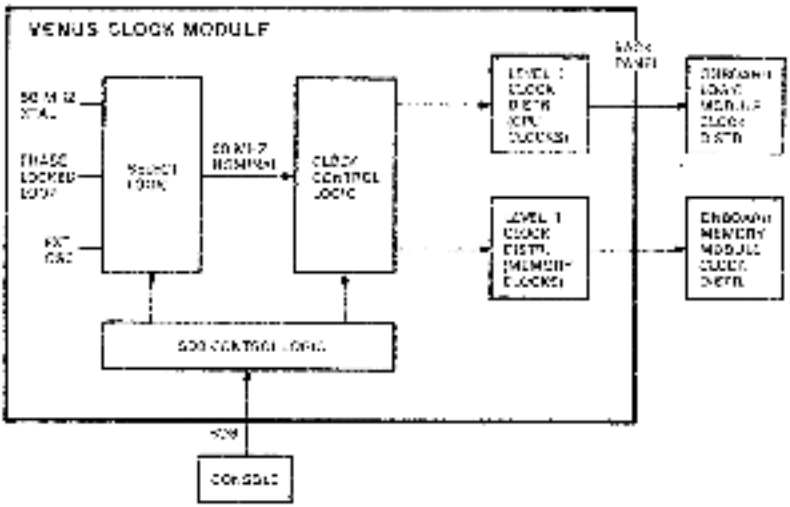


Figure B-1 Clock Distribution System, 10217 Rev. C5

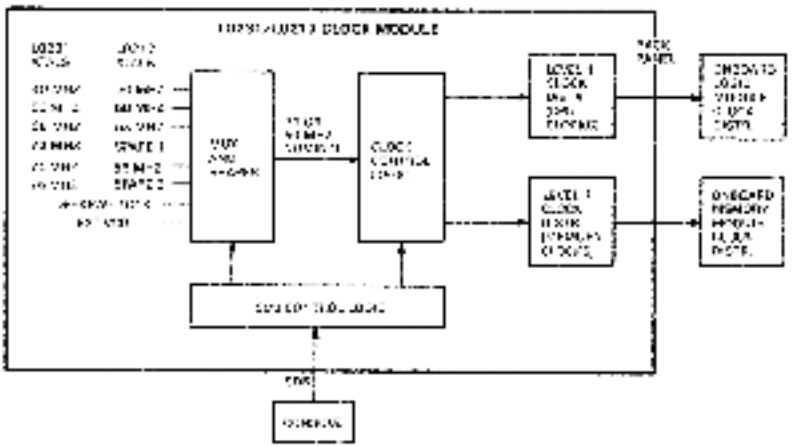


Figure 4-2 Clock Distribution System, 10231/10217 Rev. E

Figure 8-3 [021] Rev. C5 Block Diagram (Sheet 1 of 2)

Figure 8-3 L0217 Rev. C5 Block Diagram (Sheet 2 of 2)

Figure 8-4 LB231/LB217 Rev. E Block Diagram (Sheet 1 of 2)

Figure 9-1 DC231/DC237 Rev. B Block Diagram (Sheet 2 of 2)

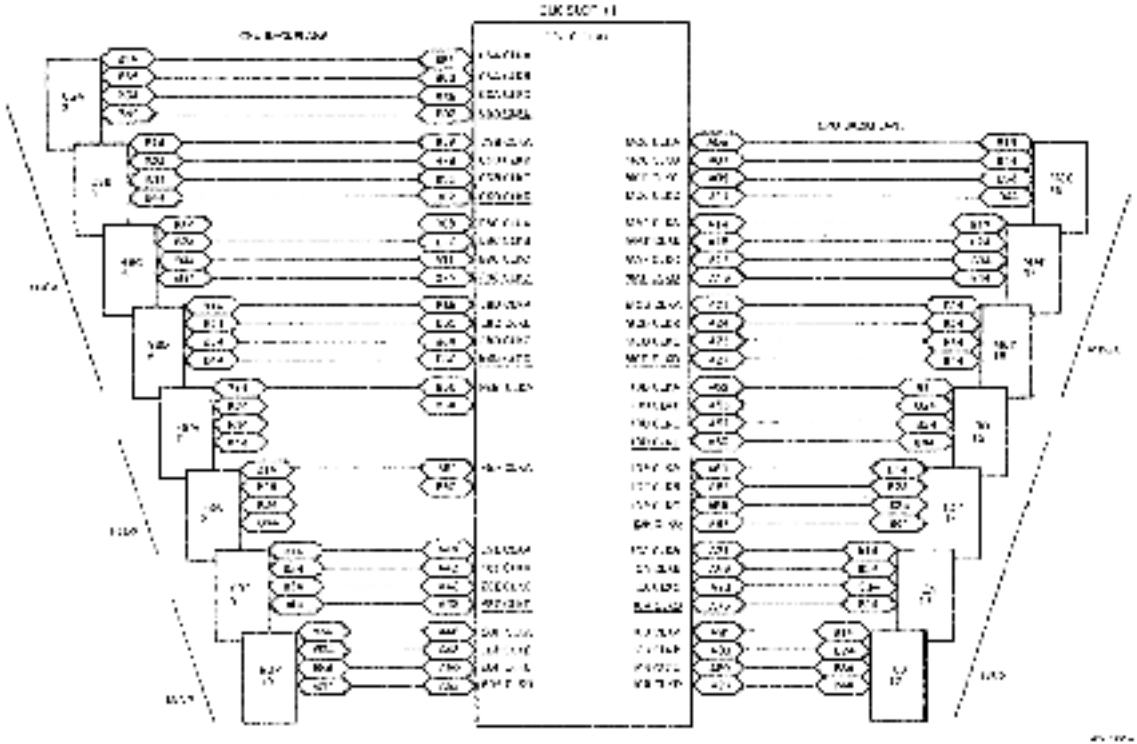


Figure 8-5 Backplane Clock Distribution (Sheet 1 of 2)

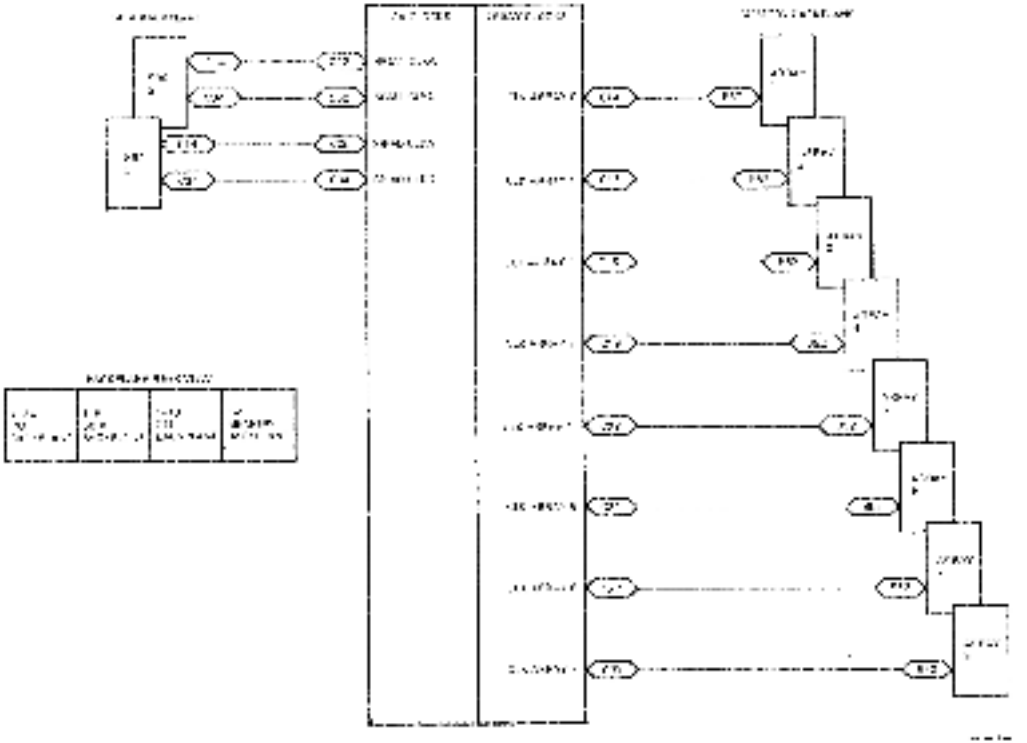


Figure 6-5 Backplane Clock Distribution (Sheet 3 of 4)

Figure 4-6 Clock Control Logic, CEC MCA

CHAPTER 9

CONSOLE HARDWARE

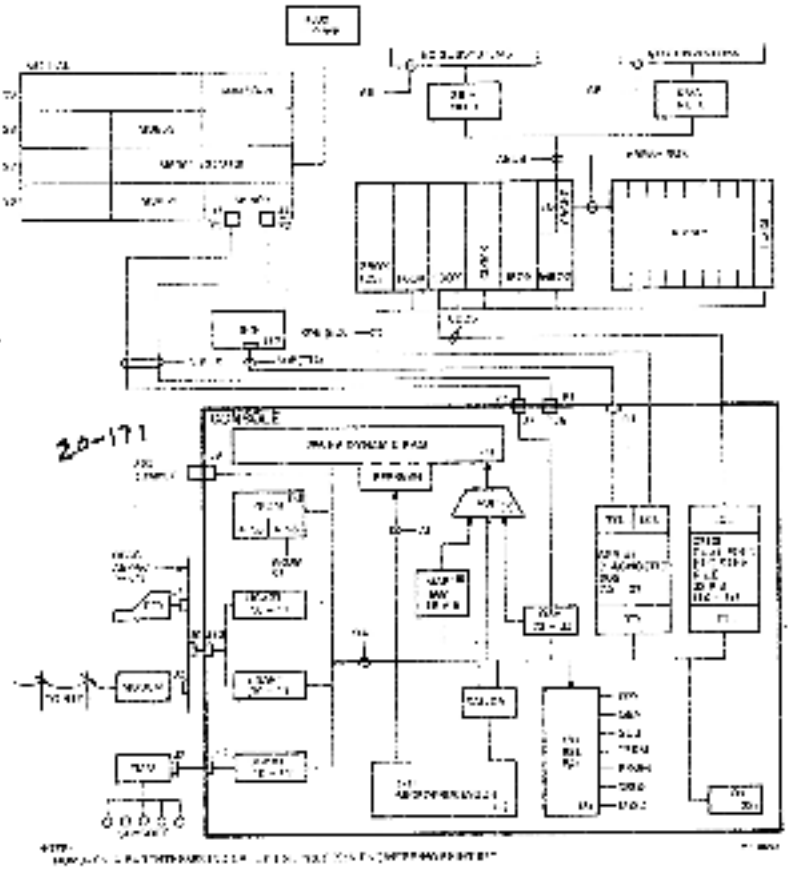


Figure 9-1 Console Interconnect Block Diagram

9.1 SYSTEM CONTROL PANEL

The system control panel (SCP), is located on the top right hand side of the CPU cabinet. See Figure 9-2. It consists of two switches and four status indicators (LEDs). The SCP communicates only with the console by means of the Serial Diagnostic Bus (SDB). The console can read the state of the switches and read and write the LED register, which is the input to the indicators. The setting of the two switches combined with the current console mode determines the response of the console program to various external inputs.

The right switch, the terminal control switch, is used to apply, or remove, DC power to the CPU and for setting the mode of operation for the local and remote terminals. The left switch, the restart control switch, is used for bootstrapping the system and controlling automatic restarts.

The status indicators show the CPU run state, the status of the remote diagnostic link, and the occurrence of an environmental (or other) fault that is detected by the Environmental Monitor Module (EMM).

9.1.1 Console Operating Modes

The console operates in one of two modes:

1. Console I/O mode (CIO): Operator command input is accepted.
2. Program I/O mode (PIO): VMS running, the console acts as a slave to the VAX processor and services its requests.

9.1.2 SCP Switches and Indicators

The control panel switches and indicators are shown in Figure 9-2. The control panel indicators are described in Table 9-1 and the switches are described in Table 9-2.

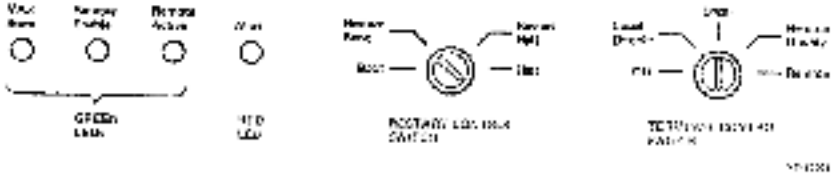


Figure 9-2 System Control Panel Switches and LEDs

Table 9-1 System Control Panel Indicators

Control Panel Indicators	State	Description
VAX State	OFF	CPU is not running, or the console program is not in PIO mode.
	BLINKING	CPU is executing a VAX program.
Remote Enable	ON	Terminal Control Switch is in the Remote or Remote Diag/Loop position. Console program is in PIO mode.
Remote Active	ON	Remote terminal is connected and the line is active.
Alert	ON	SBH has detected a fault condition, an error message has been printed on the console terminal indicating a "yellow-zone" temperature condition. LED is turned off when violation has been eliminated, as detected by the SBH.
	BLINKING	Flashes in 1 second intervals to indicate "red-zone" condition, and message is printed on the console terminal. Total system power shutdown occurs in 1 minute if condition is not eliminated. Conditions that border between yellow and normal may cause LED to appear as flashing.
	RIMBING	Flashes when an air flow fault is pending. Automatic shutdown is armed to trip in 2 minutes for a single air flow violation. Shortened for a double air flow fault (2 sensors).

NOTE

These indicator descriptions are valid only when console software is up and running. When SROM code is running, the indicators may have different meanings, see SROM Troubleshooting, Chapter 4.

Table 9-2 summarizes actions taken by the console for all combinations of the Restart Control Switch (RCS) and the Terminal Control Switch (TCS) on the System Control Panel (SCP).

Table 9-2 System Control Panel Switch Summary

Restart Control Switch	Terminal Control Switch	After Resp Alive Poll and Initialization	Remote Reset Enabled	TE HALT Enabled
Any	Local Disable	Restart Boot	No	No
Boot	Local	Boot	No	Yes
Restart Boot	Local	Restart Boot	No	Yes
Restart Halt	Local	Restart Halt	No	Yes
Halt	Local	Halt	No	Yes
Boot	Remote Disable	Boot	Yes *	No
Restart Boot	Remote Disable	Restart Boot	Yes *	No
Restart Halt	Remote Disable	Restart Halt	Yes *	No
Halt	Remote Disable	Halt	Yes *	No
Boot	Remote	Boot	Yes	Yes
Restart Boot	Remote	Restart Boot	Yes	Yes
Restart Halt	Remote	Restart Halt	Yes	Yes
Halt	Remote	Halt	Yes	Yes

* Remote access allowed in FIM mode only.

9.2 CONSOLE (T11) INTERRUPT AND VECTOR INFORMATION

Table 9-3 lists the console interrupt vectors and their relative priority.

Table 9-3 T11 Interrupt Vectors

Vector	PR	ID	Device
60			Reserved trap 0
64			Reserved trap 4
10			Reserved instruction trap
14			Breakpoint vector
20			TRT trap
24	6	00	Prom Restart (unmaskable) - EXT On Input (power fail)
30			EMT Instruction trap
34			Trap Instruction trap
60	5	01	Console DCI Transmit/Receive/Modem
64	5	02	Local DCT Transmit/Receive
70	5	03	Qbus Reply Timeout
100	6	04	Unused
104	6	05	TOY 1 Hz Interrupt
110	6	06	CPU Control Store Parity Error
114	6	07	RTC4 Ready
120	5	10	TXCS Ready
124	5	11	RXCS Done
130	5	12	Qbus Adapter
134	5	13	EMR DCI Transmit/Receive
140	7	14	Console RAM Parity Error
144	7	15	Unused
150	7	16	System AC fail
154	7	17	Qbus Dead

NOTE

1. VECTOR format:

   VECTOR location — The new PC of the Interrupt Service Routine is stored here.

   VECTOR + 2 — The new PSW is stored here. Currently, this includes the Processor priority which is set to PR7 (highest) and the TB of the vector is ORed into the least significant 4 bits.

2. Self-Test numbers 34 & 35 will set up the interrupt vectors, enable interrupts then check that no spurious interrupts occur. The object is to verify the integrity of the interrupt system prior to booting RT11.

3. When a spurious interrupt does occur, the following message will be printed on the console terminal (and control will be placed back to PROM prompt):

```
?PROM ENTRY THRU INTERRUPT VECTOR nnnnnn
?REGS ..R0....R1....R2....R3....R4....R5...
R6....PSW....PC..
PROM>
```

4. CALL START interrupt can be generated by the CPU and causes the T11 to jump directly to PROM address 172004. This restart will reinitialize the console logic and reboot the console software without affecting the CPU (basically the VMS reboot console request). Note that this is interrupt vector 24.

5. CLOCK FAIL RESTART interrupt (which also vectors to 24) is from an external input on the CPU backplane (slot 2, pin C69...grounded to assert). This interrupt, as with the POWER cannot be masked. The interrupt forces the T11 to PROM address 172010 which simply prints the T11 status and returns to the PROM null loop (prompt: "PROM>"). The console message will read as follows:

```
?PROM ENTRY THRU INTERRUPT VECTOR 000024
?REGS ..R0....R1....R2....R3....R4....R5...
...PSW....PC..
ROM>
```

6. The interrupt system is controlled by "CLOCK ENA T11 INTR H" which must be asserted before ANY of these interrupts can be seen by the T11. The signal is controlled as follows: Interrupts are enabled by setting bit <0> in MCSR0, 776040. Interrupts are disabled by clearing bit <0> in MCSR0, 776040.

9.3 CBus

The console and SBox communicate over the CBus, a backplane interconnect between the two devices. See Figure 9-3. The SBox backplane connections are listed in Table 9-5, and the signals are described in Table 9-4.

Table 9-4 CBus Signal Description

Signal Name	Description
CBUS A<5:0>s	The Console Bus interface consists of a dual port RAM which has thirty-two RAM plus four external eight bit register locations. The CBUS signal lines enables the CPU to access the entire DZ CSR's interface register space.

Table 9-4 CBus Signal Description (Cont.)

Signal Name	Description
CBUS D<7:0> H	A CPU CBUS access involves the transfer of eight bits of data. The CBUS data path is an MCU bi-directional interconnect and it is controlled by the CPU CBox.
CBUS WRITE L	When this CPU controlled signal line is set and used with the generation of a CBUS CLOCK a CBUS write access is executed. The CPU clearing of this signal, and the use of the CBUS Clock initiates a CBUS read access.
CBUS CLOCK L	This signal line is sourced by the CPU and is used in combination with the CBUS WRITE signal to enable the timed sequencing of either a read or write operation.

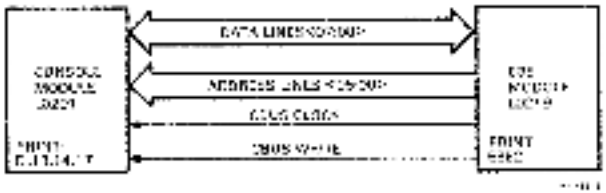


Figure 9-1 CBox Block Diagram

Table 9-5 CBus Signal Backplane Pin Location

Signal Name	CBL (D201) Slot 2	XBE (D202) Slot 9
CBUS D0 H	C62	C66
CBUS D1 H	C66	C65
CBUS D2 H	C70	C68
CBUS D3 H	C71	C71
CBUS D4 H	C61	C67
CBUS D5 H	C73	C72
CBUS D6 H	C74	C56
CBUS D7 H	C64	C62
RMT CBUS AC H	C78	C80
BBE CBUS A1 E	C76	A20
RBA CBUS A3 F	C46	C64
BBB CBUS A2 E	C76	C7A
RBC CBUS A4 L	C45	C47
BBF CBUS A5 F	C44	A74
BBC CBUS AXILE H	C52	C51
CBUS CLOCK L	C58	C56

9.6 QBUS

The console interface to the MS02 is through the QBus adapter and over the QBus to the RLV12. Figure 9-4 is a block diagram of the QBus interconnect between the console and RLV12 Disk controller. Table 9-6 lists the QBus backplane pin connections, and Table 9-7 is a description of the QBus signals. For more information, see KA86 Console Technical Description, EK-KA86C-TD.

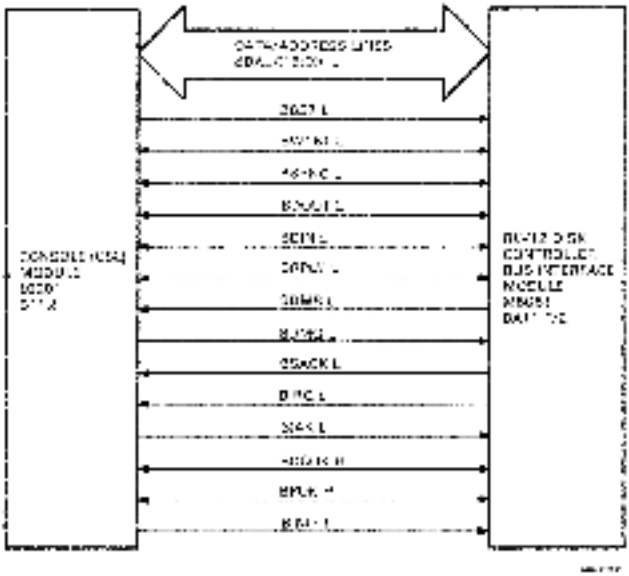


Figure 9-4 QBus Interconnect Block Diagram

Table 9-6 QBus Signal Backplane Pin Locations

Signal Name	C2U Backplane QBL Slot X	Cable	DAI Front End Cable	Backplane Slot 26
BDCOK0			J1 U	AA1
BDAL 0 L	A32	J7 04	J1 D	AD3
BDAL 1 L	A30	J7 06	J1 F	AV2
BDAL 2 L	A30	J7 0	J1 J	BB2
BDAL 3 L	A29	J7 10	J1 L	BF2
BDAL 4 L	A34	J7 12	J1 M	BH2
BDAL 5 L	A46	J7 14	J1 R	BJ2
BDAL 6 L	A41	J7 16	J1 T	BK2
BDAL 7 L	A35	J7 18	J1 V	BL2
BDAL 8 L	A05	J7 20	J1 X	BM2
BDAL 9 L	A20	J7 22	J1 AA	BR2
BDAL 10 L	A06	J7 24	J1 CC	BP2
BDAL 11 L	A12	J7 26	J1 EE	BU2
BDAL 12 L	A07	J7 28	J1 HH	BS2
BDAL 13 L	A18	J7 30	J1 KK	BT2
BDAL 14 L	A19	J7 32	J1 MM	BL2
BDAL 15 L	A14	J7 34	J1 PP	BV2
BWTBT L			J1 SS	AK2
BBS7 L	A44	J7 36	J1 UU	AP2
BDOUT L	A36	J8 04	J2 D	AB2
BREPLY L	A39	J8 06	J2 F	AV2
BDIN L	A38	J8 08	J2 J	AH2
BSYNC L	A37	J8 10	J2 L	AJ2
BIRQ L	A42	J8 12	J2 M	AL2
BIAKI L	A74	J8 14	J2 R	AK2,AV2
BINIT L	A84	J8 16	J2 T	AN1
BWTBG L	A70	J8 18	J2 V	AV2,AT2
BPOKI L	A76	J8 20	J2 X	AV2
BHALT L			J2 AA	AP1
BSPARE 3 L			J2 CC	AC1
BREF L			J2 FF	AJ1
BSPARE 4 L			J2 HH	AD1
BCLOK L	A26	J8 30	J2 KK	AA1
BPOK L	A43	J8 32	J2 MM	BS1
BSPARE 6 L			J2 PP	BP1
BSPARE L	A45	J8 34	J2 SS	BM1
BSPARE L			J2 UU	BR1

NOTE

In order to avoid confusion between QBus and console signals that have the same name but different meanings, all QBus signals are prefixed with a "B", e.g., "BSYNC" is a QBus signal and "SYNC" is a console (QBA) signal.

Table 9-7 QBus Signal Descriptions

BDAL<15:0> L	Sixteen multiplexed data/address lines.
BBS7 L	Bank 7 (I/O page) select. Asserted by console (bus master) during a DATO or DATI when address is a controller (I/O) address. (Needed for some applications in other systems when both I/O and memory devices are connected to the bus.)

Table 9-7 QBus Signal Descriptions (Cont.)

DWTBY L	Write/byte. Asserted by bus master during address portion of DATO to indicate write data is to follow.
BSYNC L	Synchronize. Asserted by bus master during DATO or DATI to indicate it has placed address on bus. The data transfer is in progress until SYNC is negated.
DOUT L	Data out. Asserted by bus master during a DATO to indicate it has placed data on bus.
DIN L	Data in. Asserted by bus master during a DATI to indicate slave may place data on bus. Also asserted by console during interrupt acknowledge to indicate controller (the interrupting device) may place interrupt vector on bus.
SRPLY L	Reply. Asserted by slave during a DATO to indicate it has taken data on bus, and during a DATI to indicate it has placed data on bus. Also asserted by controller during interrupt acknowledge (by console) to indicate it has placed interrupt vector on bus.
BDMR L	Direct Memory Access (DMA) request. Asserted by controller to request bus mastership for DMA transfers.
BDMG L	Direct Memory Access (DMA) grant. Asserted by console (bus arbitrator) to grant bus mastership to controller for DMA transfers.
BSACK L	Selection acknowledged. Asserted by controller to indicate it has assumed bus mastership following a DMA request.
BIRQ L	Interrupt request. Asserted by controller to interrupt the console.
BIAK L	Interrupt acknowledge. Asserted by the console to tell controller to place interrupt vector on bus.
BDCOK H	DC power OK. Indicates there is sufficient DC voltage to maintain reliable operation. Negated by console or controller when cabinet power supply detects DC LOW condition.
BPOK H	AC power OK. Indicates normal AC power. Negated by console or controller when cabinet power supply detects AC LOW condition.
BINIT L	Initialize. Asserted by console to reset controller to initial state.

9.5 SDH

The console uses the SDH to load and verify system firmware, and to read the state of hardware signals for diagnostics and error handling. Figure 9-5 is a block diagram of the SDH interface, Table 9-8 lists the SDH interface to the SCP, Table 9-9 lists the SDH control signal interface, and Table 9-10 lists the SDH clock and data out interface. Figure 9-5 refers to signal charts A (Table 9-8), B (Table 9-9), and C (Table 9-10).

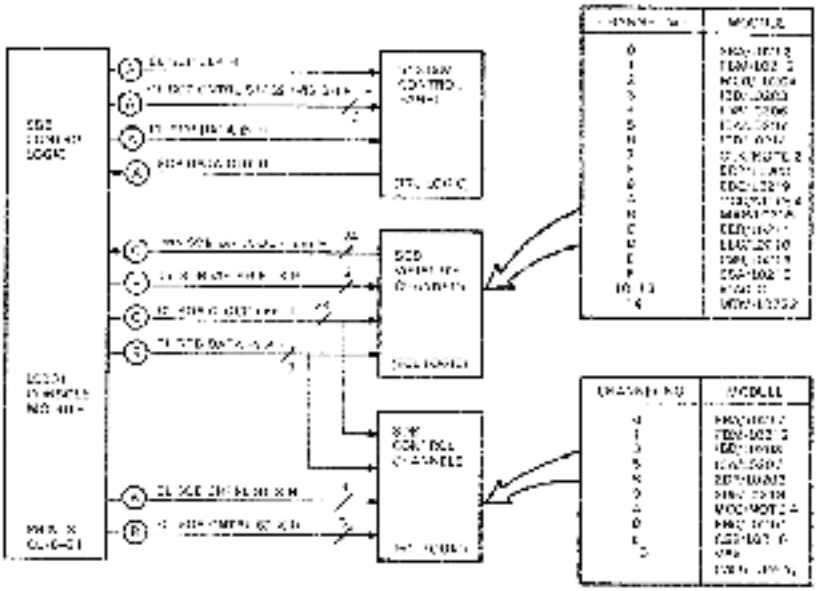


Figure 9-5 SJB Interfaces

Table 9-8 SJB Interconnect Chart A - SCP Interface

Signal Name	CBL	CPU S/P	SCP
CL SCP CTRL S1 H	B02-69	J11-04	J12-04
CL SCP CNTRL S1 L	B02-50	J11-00	J12-08
CL SCP CNTRL S2 E	B02-68	J11-10	J12-10
CL SCP DATA IN H	B02-64	J11-02	J12-02
CL SCP VIS SHIFT	B02-67	J11-14	J12-14
CL SCP DATA OUT H	B02-60	J11-06	J12-06

[illegible] [illegible] [illegible] [

Table 9-13 SDB Interconnect Chart C - SDB Clock and Data Sub Connections

Module	CL SDB CLK nnn E (CN)	Module	nnn SDB DATA OUT nnn H (CN)	Module	
L0204 MCD	B03-07	A10-07	B03-22	A16-37	
L0205 MAP	C02-06	B17-77	C02-31	C17-65	
L0206 EDP	B02-75	A10-04	B02-25	A16-55	
L0207 ICA	B02-74	C13-25	B02-52	B13-25	
L0208 IBB	B02-76	A15-50	B02-54	A15-47	
L0209 EDP	C02-06	B17-77	C02-52	A10-03	
L0210 XBC	C02-05	CC5-05	C02-32	CC5-35	
L0211 DBR	C02-07	C06-07	C02-36	CC6-16	
L0212 PMA	B02-66	A08-60	B02-63	A08-32	
L0213 PDM	B02-70	A07-91	B02-56	A07-48	
L0214 ICB	B02-72	B12-28	B02-53	C12-06	
L0215 CSA	C02-10	CC3-17	C02-40	CC3-40	
L0216 CSD	C02-11	CC4-11	C02-35	CC4-35	
L0217/					
L0221 CLK	B02-65	C11-04	B02-49	C11-87	
L0219 BBB	C02-09	CC9-10	C02-57	BC9-70	
L0220/					
L0230 MCC	C02-10	A16-81	C02-34	C18-34	
L0222 MTH	C02-29	J06-21	C02-50	J06-27	Note 2
10A0	C02-26	J14-12	C02-37	J14-42	Note 3
10A1	C02-28	J14-14	C02-42	C14-06	Note 3
10A2	C02-30	J14-16	C02-39	J14-02	Note 3
10A3	C02-32	J14-18	C02-41	J14-04	Note 3
Z1	C02-27		C02-47		Note 4
Z2	C02-20		C02-29		Note 4
Z4	C02-24		C02-60		Note 4

NOTES

1. To determine the correct signal name, replace the "nnn" with the appropriate module mnemonic. For example, on the L0206 MCD module, the signal "CL SDB CLOCK nnn H" becomes "CL SDB CLOCK MCD H".

2. The connections in the Alto backplane to the MTH module are as follows:

CL SDB CLOCK MTH H	J06-22	A09-39
MTH SDB DATA OUT MTH	J06-20	A09-73

3. These connections are reserved for the 10MA visibility modules, which are for manufacturing use only.

4. These connections are spare connections reserved for future use.

9.5.1 SDB Signal Name File Information

To support the SDB channels, Signal Name Files or CADIF files (Computer Aided Design Information Files) are required to translate a given logical SDB signal name or symbol to the physical SDB location. The following text outlines some of the components used in the visibility logic.

9.5.2 CADIF File Description

The following text is taken from a sample CADIF file: CSDB01.CDF

```
;CHASSR Version 1(31)+1, 31 January 1984. Software in [SDB4]:CSDB>
/CADIF-VERSION/ 3(5)
/SBUS=SDB/ 1(2)
$BUS-CHANNEL-TO-SIGNAL
16000                          V$R154    /DDP CNDX IS H
16001                          V$E125    BBC DIAG BX COND 1 H
16002                          V$E126    DDC TRAP ED WRI H
  :                              :       :
  :                              :       :
16345                          V$R153    FDA CARRY TO DDOY H
16346                          V$E154    BDP OCC 2 H
16347                          V$K155    BRANCH CONDITION 23 H G

$BUS=SIGNAL-TO-CHANNEL

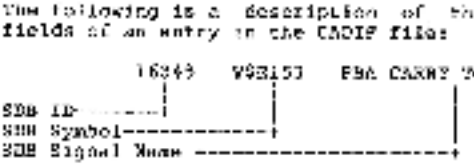
16141                          V$R101
16200                          V$E101
16201                          V$R101
16115                          V$E101
16117                          V$E103
16105                          V$E101
16107                          V$R101
16150                          V$E140    -CSDB FLIP USTK PAR H
16204                          V$R166    -CFRT CLK6 PHASE YDA G
                                         :
                                         :
16102                          V$K186     MCC TRAP UP WORK H
16043                          V$E135     MCC TRAP OP WRT E
```

NOTE

1. There is one CADIF file for each module that has a visibility channel and the file contains a list of all signals that are visible (and thus terminated) on that module. The CADIF file does not list all the signals generated from the module, rather it lists all the signals that terminate on the module.

2. The CADIF file is divided into two equal sections, a CHANNEL-TO-SIGNAL section and a SIGNAL-TO-CHANNEL section. Each section contains a complete list of all the visible signals; the first section sorted by the SDB ID, and the second section sorted by the signal name.

3. The following is a description of the three fields of an entry in the CADIF file:



4.5.1 SDB ID

This is the physical 'address' of a given signal name. The information contained within this 16-bit octal value includes the SDB channel number, bit select, VTERM MUX number and VTERM MUX module (see SDB ID Format). Every visibility point within the system has a unique SDB ID code. The SDB ID code may change from one revision module to another. The console software uses the information from the SDB ID code to program the actual SDB registers (SDDB, SDMB, and SCSR).

This field contains an OCTAL value. If the entry is 17777, it indicates that there is no physical address for a given symbol/name. This case occurs when a signal is removed from the visibility logic on a new revision of the module.

The SDB ID is not a hardware register. It is simply a 'logical' register used by the console software to identify the physical location of a given visibility point. In general, the SDB ID is broken down into two parts. The first part, the CHANNEL SELECT field, identifies which channel (or module) we want to look at. The second part, consists of the BIT SELECT and MUX SELECT and ENABLE fields, which is used to address one bit of a possible 512. It is this last portion which is implemented differently on different modules. For the current system, the following table indicates the number of visibility points per module. Refer to the appropriate technical description manual for a detailed functional description of how the SDB logic is implemented on the individual modules.

Module(s)	Maximum Number of Visibility Points Implemented
FBA, FBM	288 (12 X 24)
ICA, ICE, IBD, IDP	256 (8 X 32)
All others	192 (8 X 24)

Figure 9-6 shows the bit breakdown of the SDB ID and Table 9-11 defines the MUXs (6).

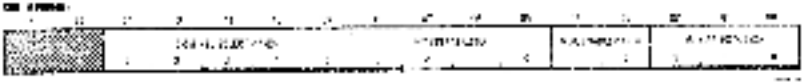


Figure 9-6 MDS ID Format

Table 9-11 SDR to Bit Descriptions

SDR Bit	Bit Description
CHANNEL SELECT <4:0>	Select one of 32 (of possible channels. Currently used channels are:

Channel	Module	Mnemonic
00	LC212	PEA
01	LC213	PBM
02	LC204	MCD
03	LC208	IPE
04	LC206	IDP
05	LC207	ICA
06	LC214	ICE
07	LC217	CLK
08	LC209	CDP
09	LC219	LBI
0A	LC220	XCC
0B	LC205	XAP
0C	LC211	XSU
0D	LC210	XBC
0E	LC215	CSG
0F	LC218	CSA
10	N/A	IOA0
11	N/A	IOA1
12	N/A	IOB2
13	N/A	IOB3
14	LC222	MTX
15-1F	RESERVED	

BIT SELECT <3:0>	Select one of 16 bits for a given bit position on the visibility channel. The PEA and PBM modules support 12 bits, while all other modules support 8. The SDR register is 8 bits wide, so the actual number of bits read must be a multiple of 8. When reading data on the PEA and PBM visibility channels, there will be 4 filler bits.
MUX ENABLE <1:0>	Select which bank of SBP visibility channel multiplexers will be matched.
MUX SELECT <2:0>	Select 1 of 8 inputs to the SDR visibility channel multiplexers.

9.5.4 SDB Symbol

The SDB symbol is the logical symbol assigned to a given signal name. The objective is to assign a permanent code to a given signal name which can be coded into the diagnostics (MAC and Micro diagnostics). By using the SDB symbol, rather than the actual signal name, less program space is required. The SDB Symbol for a given signal name will not change.

The format for the SDB symbol is: VBCNKN, where:

VB - symbolizes that this is an SDB Symbol

C - represents the SDB visibility channel number thusly:

0 = PBA	1 = PBB	2 = MCD
3 = IBD	4 = IDP	5 = IC4
6 = IC5	7 = CDE	8 = EDE
9 = SDE	A = XCC	B = MAP
C = SBD	D = SBC	E = CSB
F = CSA	G = ;CAB	H = IDA1
I = IDA2	J = IDA3	K = MCX

KNK - is a unique DECIMAL number identifying a given signal name on the module.

9.5.5 SDB Signal Name

This is the signal name which is used in the circuit schematic drawings. Although the signal polarity may change within the schematics (e.g., from MICS MAPPED H to -MICS MAPPED L) it follows a common convention in the CADIF files. All SDB signal names contain the 'H' suffix. Thus if the signal is actually a true low signal such as 'MICS MAPPED L' it will be listed in the CADIF file as '-MICS MAPPED H'.

Thus, if you are tracing through the print set using the 'EXAMINE/SDB SIGNAL NAME' command under VMB and come across a signal such as 'ICB ID FULL STALL L' and you enter the command:

```
>>>EXAMINE/SDB "ICB ID FULL STALL L"
```

You will get the following error message:

```
?SCH-W-CUSIG, signal name not found in CAD tables
```

You need to convert the signal name to the 'H' polarity thusly:

```
>>>EXAMINE/SDB "-ICB ID FULL STALL H"
    V$5196 -ICB ID FULL STALL H = 1
```

9.5.6 CADIF File Revisions and the SDBHISTORY file

It is possible that for every revision of a given module, the related CADIF file may change. Hence, the names allocated to the CAD.F files were intended to reflect the revision of the module. (For example, CSB302.CDF is the CADIF file for the CSB module, revision BC2). This, however, became the exception rather than the rule and as a result, the naming convention for the CADIF files was not followed.

To determine the correct CADIF file for any revision of a given module within the system, use Table 9-12 for the VAX 8600 and Table 9-13 for the VAX 8650. To ease the task of keeping track of the different CADIF files, the CONFIG.DAT file was implemented. This file contains a list of the .CDF files and is initially loaded by the console software. Thus, for a given system, the CONFIG.DAT file must be at a specific revision. This is taken care of via console SLU2 pack revisions. For example, if your system IBOX is at Rev. B3, then you must use a Rev. 5.0 SLU2 pack (which will have the appropriate CADIF and CONFIG.DAT files).

Note that if the correct SLU2 pack revision is used on a given revision EA86 system, it should not be necessary to investigate and/or alter these files as the console software and CONFIG.DAT file should handle the configuration.

The CONFIG.DAT file may contain a comment field for each of the CADIF files which indicates a module revision. Beware that this field is information only and may not accurately reflect the revision(s) of the module(s) supported by the CADIF file.

Table 9-12 VAX 8600 CADIF File Revision Information

Module	CADIF File	Module Revisions Supported
LO217	CLKC03.CDF	Revision C5 and earlier
LO217	CLKL01.CDF	Revision S1 and later
LO215	CEAB02.CDF	ALL
LO216	CSBB02.CDF	ALL
LO210	KNCC05.CDF	ALL
LO211	EBDD02.CDF	ALL
LO219	EVEB0A.CDF	ALL
LO209	EDPC02.CDF	ALL
LO212	FBAB01.CDF	ALL
LO213	FBNC01.CDF	ALL
LO208	IBUF05.CDF	ALL
LO207	ICAE02.CDF	ALL
LO214	ICBF01.CDF	ALL
LO206	IDFF02.CDF	ALL
LO205	MAFD02.CDF	ALL
LO020	MCCJ04.CDF	Revision J04 and earlier
LO220	MCCK01.CDF	Revision K01 and later
LO204	MCDD04.CDF	ALL
LO222	MINN01.CDF	ALL
LONBK	VBAA01.CDF	(SBIA VISIBILITY MODULE 1) M8J only
LONBK	VBNA01.CDF	(SBIA VISIBILITY MODULE 2) M8J only

NOTE

This table reflects a VAX 8600 with Rev. 5.0 SLU2 or hardware revision EA86 Rev. B3.

Table 9-13 VAX 8650 CADIF File Revision Information

Module	CADIF File	Module Revisions Supported
LC231	CLKL01.CDF	ALL
LC215	CSAR02.CDF	ALL
LC216	CSAR02.CDF	ALL
LC2 0	RBCC05.CDF	ALL
LC211	SBD002.CDF	ALL
LC219	SUR000.CDF	ALL
LC209	DSPC02.CDF	ALL
LC212	MBA01.CDF	ALL
LC213	MBC0 .CDF	ALL
LC208	IBDE05.CDF	ALL
LC207	*CA407.CDF	ALL
LC214	ICBP01.CDF	ALL
LC206	IBPF02.CDF	ALL
LC205	MAP002.CDF	ALL
LC230	MCCA01.CDF	ALL
LC204	MCDD06.CDF	ALL
LC202	MCNB01.CDF	ALL
LC8M0	VSMA01.CDF	ISBIA VISIBILITY MODULE 1) Mfg only
LC8M0	VSBA01.CDF	ICBIA VISIBILITY MODULE 2) Mfg only

NOTE

This table reflects a VAX ABUS with Rev. 1.2 RL02 or hardware revision KA86 Rev. B.

9.6 REMOTE DIAGNOSIS

9.6.1 General

This section describes the use of remote diagnosis (RD) to troubleshoot the CPU. Use remote diagnosis to help solve problems with the system that you cannot isolate at your particular site.

RD involves connecting the system to the Digital Diagnostic Center (DDC), by telephone line.

The DDC can find system failures at the device level and identify faulty operational problems over the telephone. This service allows you to run tests on the system, without having an engineer at the site.

To use RD, you must have a direct dial phone line to the computer room. This phone line must be connected to an RCA 103 (or equivalent) full duplex modem.

9.6.2 Setting up the DF112 Modem

Remote diagnosis involves setting up the DF112 modem, connecting the modem to the system and the telephone line, self-testing the modem, and using the console software to make the necessary connections to the DDC. The DDC takes over from there, running the appropriate diagnostic routines to isolate the problem.

For systems shipped to sites within the U.S. and Canada, a DF112 modem will be included with the system.

The chapter references in the following steps refer to the DF112 User's Guide (EK-DF112-UG). This guide comes with the DF112 modem. Install the modem as follows:

1. Unpack and inspect the DF112 modem (Chapter 2).
2. Set the switchpack S1 - S4 (refer to Figure 3-7) for the required modem options.
3. If you are using a dial-up modem, install the 8A4A00 048 modem module in the DF112 modem (Chapter 2).
4. Connect the standalone modem to the appropriate public telephone network service (Chapter 2).
5. Connect one end of the data terminal equipment (DTE) cable (BC22E-xx) into the DTE connector on the rear of the DF112 modem as shown in Figure 3-9. Connect the other end of the cable to the connector marked "CUSTX" on the KA66 line distribution panel (refer to Figure 3-8).
6. Set the DF112 front panel pushbuttons to the correct positions.
7. Ensure that the VAX 9000/9500 system is in CUI mode (i.e., at the ">>>" prompt) and the terminal control switch on the SCP, is in the REMOTE ENABLE position.
8. Verify that all indicator lights on the DF112, except for "TR", are on.
9. Test modem operation by making a connection using an alternate originating modem and terminal.
10. Call the SDC and request an "FD install" verify call.

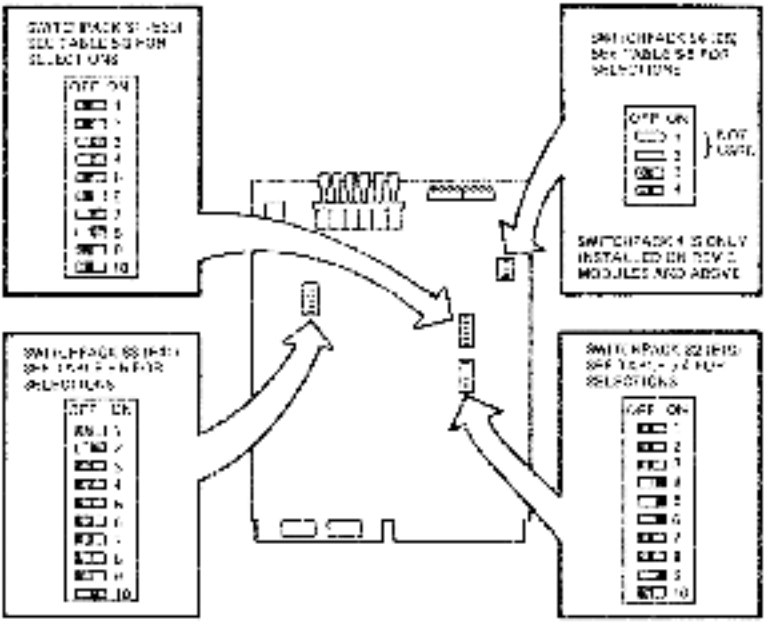


Figure 9-7 DP112 AA Switchpack Locations

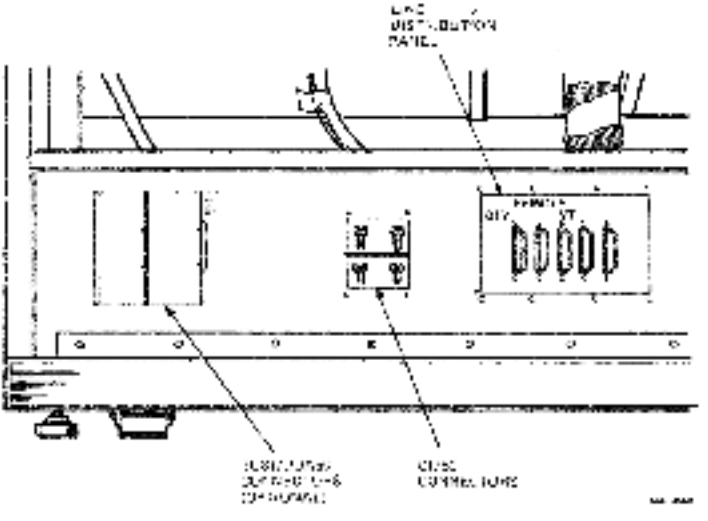


Figure 9-8 KS86 Line Distribution Panel

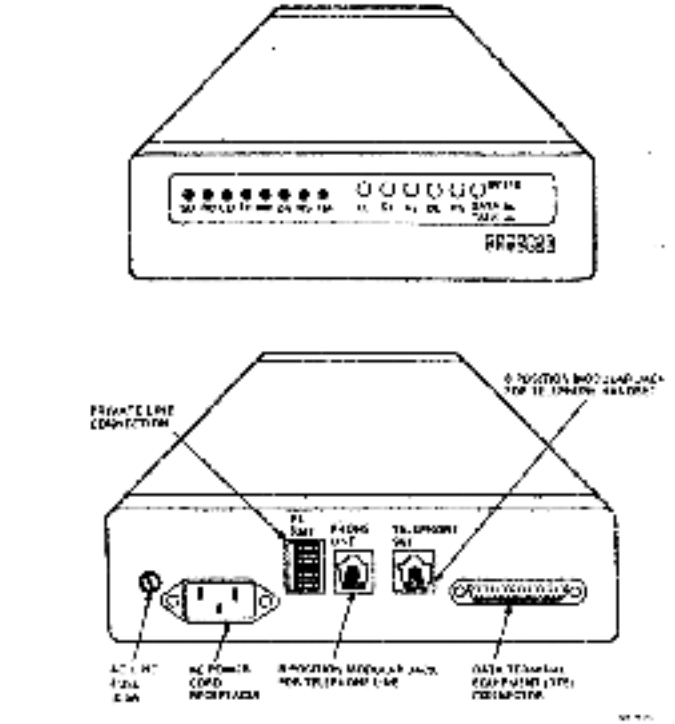


Figure 9-9 DF112 Rotary

9.6.7 Using the Set Terminal Command

Once you have completed steps 1 through 10 above, you may use the SET TERMINAL command to change the communication parameters for the modem. Use the SET TERMINAL command for the following:

1. Change the baud rate for receive and transmit
2. Enter the DDC password

The SET TERMINAL command sets terminal port characteristics for the CTY or VTY interfaces on the system. See table 9-14.

Table 9-14 Set Terminal Syntax and Switch Description

SET TERMINAL/switch, where "switch" can be any of the following parameters.

Parameter	Function
/BAU:Receive	Sets the transmit and receive baud rate for the CTY port.

Table 9-14 SET TERMINAL Syntax and Switch Description (Cont.)

Parameter	Function
/RECEIVE:nnnn	Sets the receive baud rate for the RTY port.
/TRANSMIT:nnnn	Sets the transmit baud rate for the RTY port.

NOTE

The possible values for "nnnn" in the above switches are: 50, 75, 110, 134, 150, 300, 600, 1200, 1700, 2000, 2400, 3600, 4800, 7200, 9600, and 19200.

/[NO]PARITY	Enables or disables the use of a parity bit on characters transmitted from the RTY port.
/EVEN	Selects even parity for RTY port transmissions.
/ODD	Selects odd parity for RTY port transmissions.
/[NO]DSRS	Data Set Ready Subset (DSRS) selects low speed or high speed modem operation. This switch generates an output at the RTY port on pin 23.
/[NO]SCOPE	Specifies the type of terminal device used at the RT port so that the device handles cutout characters correctly.
/PASSWORD	Sets the login password for the RTY port. The password, if used, must consist of no more than 6 characters and must contain only alphanumeric characters (0 - 9, a - z, A - Z). If no password is used, the RTY port password feature is disabled. With no password, the setting of the front panel Terminal Control Switch controls RTY access to the CPU.

NOTE

All terminal characterization set by the SET TERMINAL command are saved in console memory and lost when the console reboots or fails. Therefore, it is important that the SET TERMINAL commands be placed in the CPU initialization file, LOAD.CON

Example 9-1 Set Terminal Command

SET TERMINAL/BAUD:9600<RETURN>	Set the transmit and receive baud rates to 9600 for the RTY port.
SET/TERMINAL/TRANSMIT:4800<RETURN> SET/TERMINAL/RECEIVE:4800<RETURN>	To set the transmit and receive baud rates to 4800 for the RTY port.

After completing the above steps and using the SET TERMINAL command to set the communication environment, the CPU can now diagnose the system. See Figure 9-10 for a flowchart on control of the remote port.

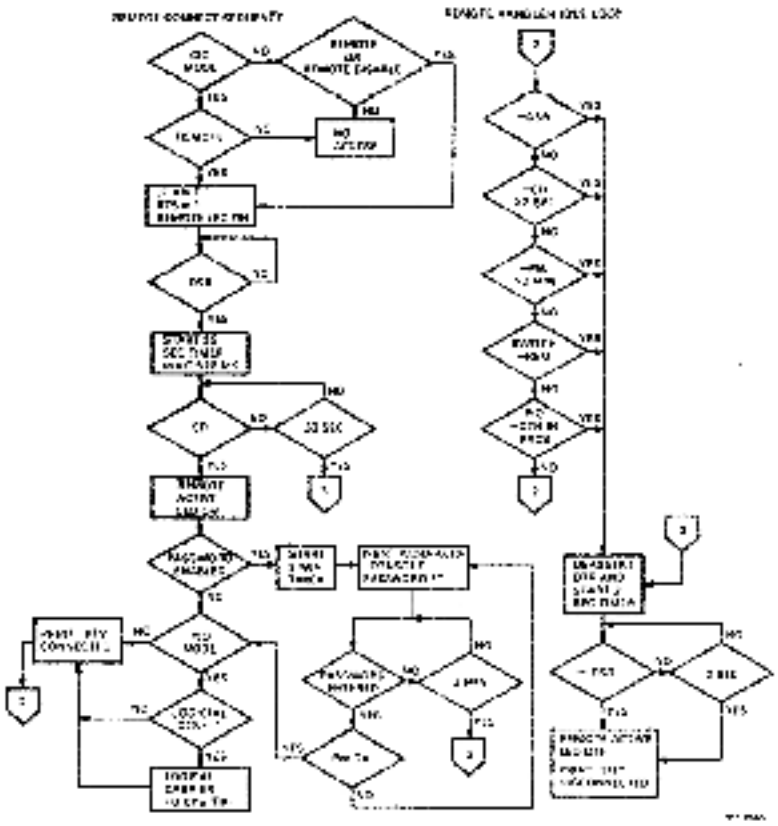


Figure 9-10 Console Remote Poll Control Flowchart

CHAPTER 10

CONSOLE SOFTWARE AND COMMANDS

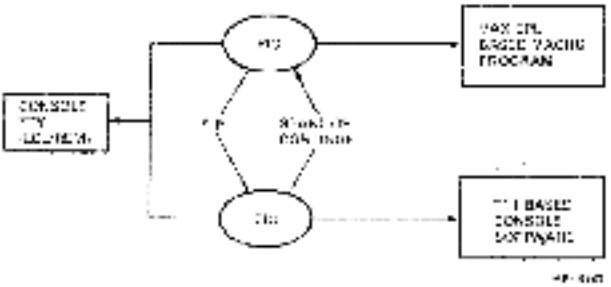


Figure 10-1 Console Software Modes

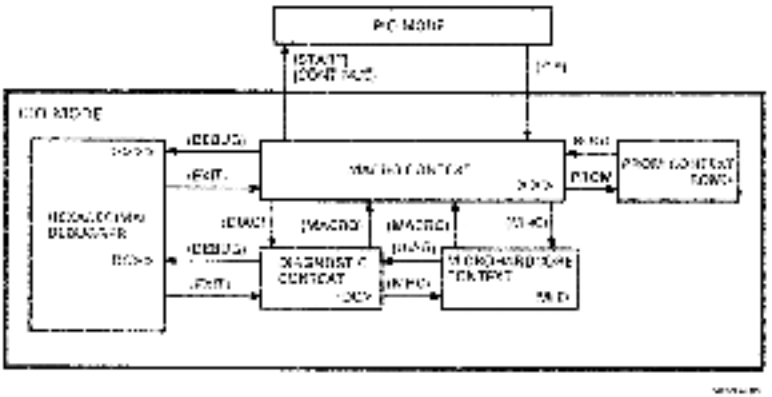


Figure 10-2 Console Mode Contexts

10.1 CONSOLE COMMANDS

Chapter 10 will include the console commands not found in Chapter 6, Diagnostics, namely: General commands, MACRO Context commands, and UNA commands. The PROM commands, MCC Context commands, and Micro-Diagnostic Context commands are in Chapter 6.

If more help is needed on any console command, use the help facility as shown in the following example.

Example 10-1: Help on Console Commands

1. If the context and command is known, type: "HELP [context] [command]", for instance, "HELP MACRO EXAMINE".

2. If you are looking for a command and are not sure what context to use, you can find out what commands are available for each context by typing: "HELP [context]", for instance, "HELP MACRO". The console will print out all of the commands within the MACRO Context.

3. If the names of the contexts escape you, then just type "HELP".

10.1.1 Control Characters

The control characters for the console commands are listed in Table 10-1.

Table 10-1 Console Command Control Characters

DELETE	Rubout last character typed
CONTROL-U	Flush Command Line
CONTROL-B	Retype command line from BUFF
CONTROL-O	Toggle output display ON/OFF
CONTROL-P	Abort current command
CONTROL-S	Abort current command

10.1.2 Console Command Syntax

The console commands use brackets ([ ]) to indicate an optional switch or keyword and braces ({ }) to show a list of choices where one item must be selected. For example, in the MACRO command: "EXAMINE [/space] [/next:hex_number] [/data_type] [hex_adr, reg_name]], the /space switch is optional and will default to /PHYSICAL if not used. The /next switch allows the examination of the next "hex_number" locations if used, and if not used, the default is to examine only the addressed location. The /data_type switch is also optional, and the default is /LONG. Within the brackets, we have to pick one of the two, either a hex_adr or a reg_name to provide the location that we want to examine.

10.1.2 Architecturally Defined Commands and Switches

Table 10-2 lists the architecturally defined commands and switches.

Table 10-2 Architecturally Defined Commands and Switches

Short Form	Long Form	Short Form	Long Form
B	BOOT	K	XKNT
C	CONTINUE	S	START
D	DEPOSIT	SK	SKT
E	EXAMINE	SH	SHOW
F	FIND	U	UNTAP
T	TEST	V	VERIFY
I	INITIALIZE	W	WAIT
L	LOAD	/P	/PHYSICAL

NOTE

The standard register names are listed under the MACRO Context DEPOSIT command. For GPRs, see Table 10-7, IPRs, see Table 10-8, and PRs, see Table 10-10.

10.2 GENERAL COMMANDS

Commands in the GENERAL COMMAND set are available in all CTC mode contexts (MACRO, DIAG, and MiniC). It provides the commands necessary to change context and control the basic console functions.

The control characters ^C and ^P are considered as be a part of the general command set. While in CIO mode, these control characters are treated as equivalent and will abort mode command lines and all levels of command files.

Table 10-3 is a complete list of the commands in the general command set with a brief description of each command. For more information, use the "HELP" command.

10.2.1 The General Command Set

Table 10-3 General Commands

BOOT	Enables the BOOT debugger command set. It is concatenated to the MACRO and DIAGNOSTIC context command sets. All trace breakpoints are cleared. See BOOT command set, Table 10-11.
DIAGNOSE	Switch to CIO mode DIAGNOSTIC context with diagnostic context initialization. See Chapter 6.
HELP	This command provides on-line help on the various console commands.
IF	IF (Cond, Cond) Cmd_string "Cond" is one or more of R80, R50, PSL, or NUMBER, and "Cmd_string" is any command valid in the current context.

If the condition(s) is (are) TRUE, then execute the command, otherwise do not execute the command.

INITIALIZE INITIALIZE [/CLOCK, /POWER, /SCB]

This command performs the initialization of three fundamental system components.

1. INIT/CLOCK - Initializes the system clock and clock distribution logic (cold reset) and sets the clock parameters to those saved with the last SET [CLOCK DEFAULT command. For console reboot, the saved parameters are lost and this command will use normal frequency, and full speed. The state of the CONFIG variable is cleared by this command.

2. INIT/POWER - This command initializes the CPU and the power system. At command completion, all voltage regulators are on with normal margins, and the CPU is monitoring regulator outputs, airflow, and temperature conditions.

3. INIT/POWER/ALTVIN - An optional form of the INIT/POWER command, it is used to adjust the CPU voltage and red zone temperature limits to account for systems at sites considerably above sea level. 'n' is the site elevation in feet.

4. INIT/SCB - This command forces all SCB control channels to the state necessary for normal system operation. The control channels are loaded as follows:

   a. CSR = 0008; -ELIE abrk Pwr a

   b. DT2 = 0050; -PIIE IDRA 4, -PLTP GDBB 1

   c. EBC = A000; Set DIAG register to 0

   d. EBC = C000; Set EIE register to 0

   e. EBC = E000; Set control to 00

   f. FSM = C000; Normal operation

   g. FSM = C000; Normal operation

   h. IBD = 1300; Normal operation (Optimize)

   i. ICA = 0000; Normal operation

   j. MCC = 0000; Normal operation

   k. VBA = C801; Normal S/DIA operation

LOAD [/S:] LOAD [/Switch] [filename [.BPK]]

'Switch' can be any of the following: /ACCESS, /CONTEXT, /CYCLE, /DCS, /ENGCN, /IPROC, /IRAM, /ITB, /IUMAP, /MCC, or /MCS and 'filename' is the name of the .BPK file to load. If no /filename is specified, the default system microcode will be loaded.

This command loads the specified file (.BPW) from the console load media to the specified control store or RAM. Not all control stores can be loaded without having been initialized with the INIT/MICRO console command (see MACRO context commands). Use INIT/MICRO command load KA86R.BKV to load the default system microcode, see Table 10-4.

Table 10-4 Default System Microcode

RAM	VAX 8600	VAX 8650
ACCESS	ACCESS.BPW	ACCESS.BPW
CONTEXT	CTX.BPW	CTX.BPW
CYCLE	CYCLE.BPW	CYCLE.BPW
EC2	KA8600.BPW	KA8650.BPW
FBACP	FADD0.BPW	FADD5.BPW
FMULS	FMUL.BPW	FMUL.BPW
FDRAM	FADD0.BPW	FADD5.BPW
IBX	IBOX.BPW	IBOX.BPW
IDRAM	KA8600.BPW	KA8650.BPW
MCF	MCF.BPW	MCF.BPW
MCS	UCODE0.BPW	UCODE5.BPW

If the specified control store has already been loaded, and has not been modified, the command will terminate without reloading the control store.

Example 10-2: >LOAD/FCS KA8650

LUPC

LUPC /Switch hex_address

"Switch" is one of the following: /SCS, /FCS, /MCS, /FRAMS, or /FDMCS.

This command will force the specified microsequencer to the address specified by hex address.

MACRO

This command switches to the CIO mode MACRO context. The MACRO context will perform its own initialization then prompt for command input. See Table 10-6 for MACRO context commands.

MBC

This command switches to the CIO mode MICROCHARACTER CONTEXT. MbC will perform its own initialization, then prompt for command input. See Chapter 6 for MBC commands.

ODT

The Octal Debugging Tool, ODT, is invoked with this command. The ODT prompt is an asterisk, *. The ODT command set is similar to that of the standard DEC ODT products.

PROM

PROM [/RT [filename]

The PROM command will pass control to the console PROM. The user is asked to confirm the request before control is changed. This command is valid from the ROM, and the PROM will continue to service the SIY.

If the /RT switch is provided to allow an entry to the RT monitor. It is for INTERNAL use only and is not recommended nor supported.

REBOOT This command will reboot the console software.

The console will follow its normal power-up procedure and attempt to remember FIO mode, if the REBOOT command was issued while in the MACRO context, and if the CP was running at the time.

When issued from the DIAG or SBC contexts, this command will not attempt to re-enter FIO mode.

REPEAT REPEAT [Dec_num] command

"Dec_num" specifies the number of times the command is to be repeated. If not specified, the command will be repeated indefinitely, or until a ^C or ^P is entered.

Commands requiring user input cannot be repeated. The ^O character may be used to bypass the output, causing the repeat function to run faster.

RESET This command performs a CPU logic initialization sequence as follows:

1. The CPU clock is stopped and state-swapped to the T3 state.
2. The signal "CLOG CPU RESET H" is asserted.
3. The signal "CLOG HOLD STATE RESET H" is asserted.
4. The CPU clock is burst at a frequency of 60 MHz for 1024 steps.
5. The CPU clock is forced to the T3 state.
6. The signals "CLOG CPU RESET H" and "CLOG HOLD STATE RESET" are deasserted.

RESTART This command will force a console program initialization and a restart of the console program, without reloading the console software. A REBOOT command is needed to reload the console software. At the end of the initialization, control is passed to CIO mode MACRO context, followed by the MACRO context initialization.

SET BASE SET BASE hex_number

The base, a 32-bit value, is added to both virtual and physical memory addresses during E/V, D/V, X/P, or J/P commands.

SET SET flag [ON/OFF] [INVALID, /OVERRIDE, ANY]

"flag" can be any of the following: ABORT, AUTO, BDO, COLD, DART, FDDX, IOSAFE, LOCAL-COPY, MEMORY, SMAP, STXALT, WARM, or ZOIST. This command controls the setting of software and hardware flags in the console subsystem.

Setting the COLD, WARM, CRSH, and DIAGI flags are applicable to the MACRO context only and result in an immediate action by the console program. The ABORT and QUIET flags control the handling of errors and command line echoing during a command file. The EXTI (external interrupt) flag and the ABUS flags directly control hardware signals on the console module. Use the SHOW FLAG command to examine the state of all flags. The /INVALID, /NOVERIFY, and /ON options apply to the DIAG flag only.

During a console reboot, the ABORT flag is forced ON, and the COLD, ERCK, and WARM flags are forced OFF. The remaining flags are restored to their original state or are not affected.

Table 10-5 lists the flags and a brief description of each.

Table 10-5 Console Flags

1. ABORT - This flag is set ON during console program initialization or whenever the console is rebooted.

   The ABORT flag provides a limited control over the error handling within a command file. Command file execution will be aborted upon detection of an error if the ABORT flag is on.

2. ABUS - This flag is set OFF during console program initialization and is unaffected by reboots.

   The ABUS flag controls the signal "CLCO ABUS ENABLE". If this flag is on, the ABus is enabled. The transition from OFF to ON causes an ABus INIT pulse to be generated on the ABUS. The INIT/POWER command generates the ABus INIT sequence and leaves the ABus enabled.

3. BBU - The BBU flag is set ON during console initialization and by the INIT/POWER command, and is unaffected by a console reboot.

   When the BBU flag is on, the battery backup unit is enabled to provide power to the system if ac power is lost.

4. COLD - The COLD flag is set ON by the console program during a CPU bootstrap. The flag is set to OFF when the bootstrap completes with success.

   The COLD flag is used to inhibit repeated attempts at unsuccessful CPU bootstraps.

5. EXTI - The EXTI flag controls the enabling of external interrupts, interrupts from the EBox to the console. When ON, the console will receive the interrupts from:

a. BBR CPU ERR (control store parity errors)

b. Addr SEAL

c. ENM (CPU AC LO

The DMCI flag is set ON by the "KII/CPC command and remains in this state during PIO mode operation. In CIO mode, this flag is turned off if any of the three CBus interrupts occur.

6. PBOX - This flag DOES NOT turn the PBus on or off. This flag controls the action of the IKII, INIT/CPU, or TNIT/PARK commands. If the PBox flag is on, these commands will enable the PBus. If the PBox flag is off, these commands will disable the PBus.

This flag is set to OFF during console program initialization.

7. IOSAFE - The IOSAFE flag controls a software write protect feature for BIX transfers from the CPU. When OFF, the CPU can modify the RIO2 pack.

This flag is set to OFF during console program initialization and its state is preserved through console reboots.

8. LOCAL-COPY - With this flag on, all RTY activity is logged on the CIY. The flag is set OFF during console program initialization, its state is preserved during a console reboot, and the RTY cannot set this flag off.

9. MEMENA - This flag controls the state of "CLOG XEM .DAT ENABLE", which, when off, disables write access to the arrays and switches all array modules to their internal refresh state.

This flag is set ON by the console software during CPU initialization and normally remains on until a power failure or system shutdown occurs.

10. SNAP - This flag is set to OFF during console program initialization, and its state is preserved during console reboot.

When the SNAP flag is on, the console will perform a CPU snapshot whenever the CPU stops executing macro instructions. SET SNAP ON/SOFTONLY may be used to disable control store and PAMN verification during the snapshot, thereby providing a quicker snapshot procedure.

SET SNAP INVALID is used to invalidate both SNAP1.DAT and SNAP2.DAT snapshot files on the RL02. This command has no effect on the SNAP flag.

SET SNAP NOW may be used to force the creation of a snapshot file based upon the current machine state. If both SNAP1 and SNAP2 are valid, this command will rename SNAP2 to SNAP1 and create a new SNAP2.

11. QUIET - When the QUIET flag is ON, all input taken from a command file will not be echoed to the CTY, only output will be displayed. When OFF, all commands in the command file are echoed on the CTY before being executed.

    This flag is set OFF during console program initialization, and its state is preserved during console reboots.

12. STALT - This flag is intended to be used by manufacturing and is not for customer use. It is used to direct CPU data to the alternate console disk, Unit 1. When this flag is on, all console disk access will be to RL02 #0, while all CPU traffic will be to Unit 1.

    This flag is set to OFF during console program initialization, and its state is preserved during console reboots.

13. WARM - The WARM flag is set OK by the console program when a CPU restart is being attempted. The CPU sets this flag OFF when the restart is successful. The purpose of the WARM flag is to inhibit repeated attempts at unsuccessful CPU restarts.

14. SET CLOCK - For Rev E01 and above clock modules

    a. SET CLOCK Xn Dec_num [[/NORMAL, /HIGH]] - This command assigns the "Dec_num" (clock frequency in MHz, 40 to 65 for a VAX 8600 and 40 to 90 for a VAX 8650) to the specified crystal mnemonic, X1, X2, etc., where the mnemonic can then be used to set the clock frequency. Any of the Xn mnemonics can be assigned as the normal or high clock rate with the /NORMAL or /HIGH switches. These assignments should be done once in CLOCK.COM. The Xn assignment is preserved during console reboots.

       Example 10-3: SET CLOCK

```
SET CLOCK X1 40
SET CLOCK X2 50
SET CLOCK X3 60
SET CLOCK X4 72/NORMAL
SET CLOCK X5 74/HIGH
SET CLOCK X6 76
```

    b. SET CLOCK FREQUENCY [NORMAL, HIGH, X1, X2, X3, X4, X5, X6, EXTERNAL] - This command is used to select one of the previously assigned values (X1, X2, etc.) as the system clock frequency, or the normal and high margin values can be selected with the keywords NORMAL or HIGH.

EXTERNAL can be used to specify that the clock be taken from the external input backplane pin. The SET CLOCK XN "Dec num" command must have been executed prior to this command.

c. SET CLOCK [FULL, ONE-FIFTH, DEFAULT] - This command sets the clock rate to full or one-fifth execution speed. Or, it forces the console to move the current base frequency and FULL/ONE-FIFTH information to be used by subsequent INIT/CLOCK commands as the default setting.

15. SET CLOCK - For Rev C03 clock modules

a. SET CLOCK FREQUENCY [Dec_num, NORMAL, QUIET] - This command selects the source of the CPU clock frequency source, a 50 KHz crystal (NORMAL), or a variable control oscillator (VCO), with frequencies in the range of 40 to 64 KHz (Dec num). The QUIET argument can be used to silence the warning messages that may occur at some VCO settings.

b. SET CLOCK [FULL, ONE-FIFTH, DEFAULT] - This command sets the clock rate to full or one-fifth speed, and forces the default clock settings to become the present values used by the INIT/CLOCK command.

SET SCMM SET SCMM [/Switches] [ON OFF]

SCMM means Stop On MicroMark, and switches can be any combination of the following: /ECS, /ICS, /MCS, or /FIFO. If no switch is specified when turning SCMM on, the default is /MCS. If no switch is specified when turning SCMM off, all SCMM enables are turned off.

This command controls the enabling and disabling of micro-mark breakpoint detection in the CPU clock module. When enabled for one or more control stores the CPU clocks are stopped in the T0 state when a mark breakpoint is encountered. A microcode mark bit has to have been previously set with the DEPOSIT/MARK command. See DEX commands, Table 10-11.

When the micro breakpoint is detected, the current UPC of the EBox, IBox, and MBox is displayed. If the clock rate was FULL, the UPC data will be 2 microcycles past the breakpoint microinstruction. If clock rate is One-fifth, the UPC data will match the breakpoint micro address, and the micro instruction will have already been executed. For this case, it is necessary to SET SCMM OFF to proceed with a TRIC or DSTAVE.

All SCMM flags are initialized to the OFF state during console program initialization and by the INIT/CLOCK command.

Example 10-4: SET SCMK/ECE OX

SET TERMINAL SET TERMINAL /Switches

The defaults shown are the ones set during console software initialization.

Switch	Initially set to
/BAUD:nnnn	As set in STARTP.COM
/RECEIVE:nnnn	1200
/TRANSMIT:nnnn	1200
/[NO]MAKE [password]	No password Jolin30
/[NO]SCOPE	SCOPE
/[NO]PARITY	NOPARITY
/[NO]ECHO	NOECHO
/EVEN	Default if parity is enabled
/ODD	

The value of "nnnn" can be: 50, 75, 110, 134, 150, 300, 600, 1200, 1800, 2000, 2400, 3600, 4800, 7200, 9600, or 19200.

This command sets the terminal port characteristics for the CTY and RTY interfaces. While in CIO mode the most significant bit of characters received from the CTY or RTY is stripped off, while in BIO mode, full 8-bit character handling is supported.

The default for the RTY is 1200 BPS transmit/receive, 1 stop bit, no parity, no echo, SCOPE, and no password.

The /BAUD switch applies to the CTY while all other switches apply to the RTY.

The RTY can enter only the [NO]SCOPE switch. All other switches must be issued from the CTY. The CTY [NO]SCOPE flag can only be done by modifying the STARTP.COM command file and rebooting the console so that RT executes the command file. By default, the CTY is SCOPE.

SHOW CLOCK This command displays the current state of the CPU and system clock.

SHOW FILE SHOW filename[.TYP] [/ASCII, /BINARY] This command displays the selected file on the console terminal in a binary format (default) or in a straight ASCII format if the /ASCII switch is used. The /BINARY switch can only be issued from the RTY with the remote protocol running. It is meant as a means for transferring binary files to a remote site.

Example 10-5: SHOW SHAP1.DAT/ASCII

SHOW STATUS This command displays the current state of the console program control flags.

SHOW PANEL SHOW PANEL [/TEST]

This command displays the current settings of the SCP switches and indicator LEDs. The /TEST switch causes the command to loop while shifting a floating "0" through the LEDs and reading back each pattern to verify the LED logic. Changes in either SCP switch will be included in the updated display.

SHOW POWER This command displays the current status of the PMR, power cycle, and of the CPU cabinet environment.

SHOW TERMINAL This command displays the state of the CTY and any ports.

SHOW UCODE This command displays the state of all control stores and RAMs as follows:

1. The current (last loaded) .BIN file name.
2. The default .BIN file name for that control store.
3. The UCODE revision of the currently loaded microcode.
4. The RAM status as follows:
   a. Bit 15 - If it equals a "1", the contents of the control store has been modified since its last load.
   b. Bit 14 - If it equals a "1", the VERIFY command has detected one or more discrepancies in the control store data.
   c. Bit <13:00> - These bits indicate the parity error count, the total number of detected parity errors for the control store, whether they were corrected or not. The MCB, CIA, CXC, and ACC RAMs always show "0" in this field.

SHOW VERSION This command shows the revision information for the following items:

1. The console program major revision
2. The revision of the console EPROM code
3. The revision of the BMB EPROM code
4. The major revision of the system microcode
5. Various address of interest to developers and debuggers
6. System ID register (Hex value and broken down by fields)

START CPU
STOP CPU These two commands provide ON/OFF control of the CPU clock. If the system clock is off, then the START CPU command will start it before starting the CPU clock. The STOP CPU command stops the CPU clock but does not affect the system clock.

START SYSTEM
STOP SYSTEM

These two commands provide ON/OFF control of the SYSTEM clock. If the CPU clock is on, then STOP SYSTEM will stop it before stopping the system clock. The START SYSTEM command starts the system clock running at the frequency last set by the SET CLOCK FREQUENCY command.

CHCHANG

The CHCHANG command will reset the CPU without clearing latched error status or affecting the system cache. The command performs the CHCHANG by:

1. Stopping the CPU clocks
2. Asserting the console signal "CLOG HOLD STATE RESET"
3. Bursting the CPU clock 1024 ticks
4. Deasserting "CLOG HOLD STATE RESET"
5. Loading the CSRs with 1000
6. Starting the CPU clock

VTERM

VTERM [Symbol_name, Hex_ID]

Symbol_name is the VS symbol assigned to the visibility signal by the SBA CADIF (.CDF) files, and Hex ID is a 16-bit SDB ID. SDB IDs are hardware visibility addresses and are defined in the .CDF files.

This command provides a fundamental aid for isolating faults in VTERM and visibility logic. The command accepts a VS symbol or SDB Hex_ID and sets up the visibility control logic to select the specified bit. The visibility bit is left selected so that static measurements can be made. Only a console RESTART, REPEAT, or another VTERM command will reset the state of the SDB control logic.

WAIT

WAIT [Hex_count]

This command will cause the console to stop processing commands for "Hex_count" number of 20 millisecond intervals. It is used in command files to introduce delays.

X

X Hex_adr Hex_count

Hex_adr is the address of main memory where data is to be transferred to/from.

Hex_count is a 32-bit integer count of the number of bytes to be transferred. Bit 31 implies the direction of the data transfer. If a "1", read memory.

Example 10-6: X command examples

1. X 100 FF writes FF bytes into main memory starting at location 100

2. X 100 0C0000s reads 12 bytes from main memory at location 100

This command is for automatic communication between the console and other systems. It is used to load (write) and unload (read) the contents of main memory. Refer to DSC #10 032.

@ @ filename[.CCM]

The @ (at) command causes command input to be taken from the file indicated by "filename". The commands may or may not be echoed on the CRT depending upon the QUIET flag. Outputs generated from the commands are always displayed.

Command files may be nested up to four levels deep, but at that level, no command can be executed that specifies access to a file on the disk (such as !DIR filename).

10.3 MACRO CONTEXT

MACRO context is the default context after a power-on or console reboot. MACRO context provides console commands to initialize the machine as a VAX processor. It also allows access to VIO mode operation of the console, which allows VAX macrocode to be run while the console acts as an interface for both console terminal and the EVM.

10.3.1 MACRO Context Initialization

Whenever MACRO context is entered MACRO context initialization is performed. If NEC or DIAGNOSTIC context is desired, MACRO context initialization may be aborted by the ^C or ^F characters. If the initialization is in response to a power-on, wait until "Initializing CPU" has been printed on the CRT before aborting the MACRO context initialization.

The SYS3 file "SYSAD.CSM" performs a complete initialization of the CPU for use as a macrocode processor. Below is a brief description of the MACRO context initialization.

1. "RESET" performs a master reset of the system.
2. INIT/POWER initializes or ensures that the power system is fully operational.
3. INIT/SDR places all SDR control channels in a known state to allow microcode loading.
4. Using SYSAD.CSM, INIT/MICRO loads all control stores with the default system microcode.
5. INIT/SDR command is again executed to force all SDR control channels to a NOP or SCN state.
6. INIT/PAMM will initialize the PAMM.
7. Unconditionally clear cache and perform an UNCAN on the IO adapters found on the system.
8. If this is the first initialization after a power-on and the battery backup unit was not active, or if this is a context switch from DIAG or NEC contexts, then clear main memory (CLEAR MEMORY).
9. Monitor the terminal control and restart control switches to determine whether to do a warm start, cold start, or enter the CLI poll loop.

10.3.2 Console Support Microcode (CSM)

Many of the functions of MACRO context depend upon Console Support Microcode (CSM), a microcode package which allows the console program to communicate with the CPU. Part of CSM microcode is contained in the normal CPUx microcode. CSM is also divided into microcode overlays which are loaded into CPUx control store by the console in response to typed commands. The console and CPU pass data packets over the CBus interface.

10.3.3 MACRO Context Command Set

Table 10-6 lists the MACRO context commands and a brief description of each command.

Table 10-6 MACRO Context Commands

BOOT	BOOT [/switches] [device] "Switches" can be a combination of: /R5:hex_num specify value to place in R5, default is 0 /NOSTART specify that the boot command file not start the operating system "Device" is a one to three character device mnemonic, such as DDO or CSI. This is expanded with "BOO.COM" to locate the boot file, as DUOBOO.COM. If "Device" is more than three characters, it is taken as a filename with the default extension .COM. This file is used to boot the operating system. If "device" is not specified, DSPBOO.COM is the default.
CONTINUE	If the CPU is running, but was halted by a ^P, a CONTINUE will cause a transition from CIS mode to PIS mode. If the CPU is not running then the CONTINUE command performs an ISCPP PLUSB and forces instruction execution to begin from the current PC.
CLEAR MEMORY	CLEAR [MEMORY] the first block of contiguous array memory, as determined by the contents of the DAKM, is cleared.
DEPOSIT	DEPOSIT [/space] [/:numeric_num] [/data_type] [hex_num, reg_name] hex_data "Space" can be any one of the following. The default is /PHYSICAL. /SCRATCH access EBox scratch pad RAM space /GPRPEAK access VAX processor GPR register space. See Table 10-7 /IPRIBKNA_ access VAX processor IPR register space. See Table 10-8 /DAMX access the DAMX RAMs /PHYSICAL the default, access VAX physical memory space /P access I-11 DAM address space /VIRTUAL access VAX virtual memory space if the memory mapping is enabled, otherwise access physical address space and ignore address bits 30 and 31. "Data_type can be any one of the following if the /space access is for /PHYSICAL, /VIRTUAL, or /P. /BYTE access a single byte of data /LONG the default, access a longword (4 bytes) of data /WORD access a word (2 bytes) of data

"hex_adr" is the numeric address of the register or address being deposited. Any of the following positional operators can be used to represent a "hex_adr" relative to the last address accessed.

The default address is set to 0 by the INIT command.

^ Access last specified address

+ Access the address following the last specified address

- Access the address preceding the last specified address

@ Use the contents of the last specified address as the address to be accessed

"Reg_name" is a valid register name for GPRs (see Table 10-7), IPRs (see Table 10-8), Internal Registers (IR, see Table 10-9), or miscellaneous registers (see Table 10-10).

"hex_data" is the hexadecimal data to be deposited into the specified address space.

NOTE

To deposit to Fll space, the address and data must be preceded by a "%O" (percent Octal) to define the address/data as octal instead of hexadecimal, the default.

Table 10-7 Supported GPR Names

Name	Access	G Address	Purpose
R0	E/D	00	Processor Register 0
R1	E/D	01	Processor Register 1
R2	E/D	02	Processor Register 2
R3	E/D	03	Processor Register 3
R4	E/D	04	Processor Register 4
R5	E/D	05	Processor Register 5
R6	E/D	06	Processor Register 6
R7	E/D	07	Processor Register 7
R8	E/D	08	Processor Register 8
R9	E/D	09	Processor Register 9
R10	E/D	0A	Processor Register 10
R11	E/D	0B	Processor Register 11
AP	E/D	0C	Processor Argument Pointer
FP	E/D	0D	Processor Frame Pointer
SP	E/D	0E	Processor Stack Pointer
PC	E/D	0F	Processor PC

Table 10-6 Supported IPR Names

Name	Access	I Address	Purpose
KSP	E/D	00	Kernel Stack Pointer
ESP	E/D	01	Exec Stack Pointer
SSP	E/D	02	Supervisor Stack Pointer
USP	E/D	03	User Stack Pointer
ISP	E/D	04	Interrupt Stack Pointer
P0BR	E/D	08	P0 Base Register
P0LR	E/D	09	P0 Length Register
P1BR	E/D	0A	P1 Base Register
P1LR	E/D	0B	P1 Length Register
SBR	E/D	0C	System Base Register
SLR	E/D	0D	System Length Register
PCBB	E/D	10	Process Control Block Base
SCBB	E/D	11	System Control Block Base
IPL	E/D	12	Interrupt Priority Level
ASTLVL	E/D	13	AST Level
SIRR	D	14	Software Interrupt Request Register
SISR	E/D	15	Software Interrupt Summary Register
ICCS	E/D	18	Interval Clock Control
NICR	D	19	Next Interval Count Register
ICR	E	1A	Interval Count Register
TODR	E/D	1B	Time of Day Register
RXCS	E/D	20	Receive Transfer Control Status
RXDB	E	21	Receive Transfer Data Buffer
TXCS	E/D	22	Transmit Transfer Control Status
TXDB	D	23	Transmit transfer Data Buffer
ACCS	E/D	28	Accelerator Status Register
MAPEN	E/D	38	Memory Management enable
TBIA	D	39	Translation Buffer Invalid All
TBIS	D	3A	Translation Buffer Invalid Single
PMR	E/D	3C	Performance Monitor Enable
PMR	E/D	3D	Performance Monitor Register (same as PME)
SID	E	3E	System ID
PAMACC	E/D	40	Physical Array Map access
PAMLOC	E/D	41	Physical Array Map Location
CSWP	E/D	42	Cache Sweep
MDECC	E/D	43	MBox Data ECC
MENA	E/D	44	Mbox Error Enable
MDCTL	E/D	45	MBox Data Control
MCCTL	E/D	4n	MBOX MCL (IPR)
MERG	E/D	47	MBox Error generation
CONT	D	48	Console Restart
DFI	D	49	Diagnostic Fault Insertion Register
EHSR	E/D	4A	Error Handling Status Register
STXCS	E/D	4C	Storage Transfer Exchange Control Status
STXDB	E/D	4D	Storage Transfer Exchange Data Buffer

Table 10-9 Supported IPR Names

Name	Access	Purpose
PBR	R	PBox Current PC Register
CBECTL	R/W	KBox Cache Control Register
CSLAN	R	Control Store Error Status Register
CSLINT	R/W	Console Interrupt status Register (PBox)
PMCS	R/W	PBox Control Status Register
EDMC	W	EBox Diagnostic Maintenance Control Register
EDPSR	R	EBox Data path status Register
EMD	R	EBox MME Register
ESRSAV	R	IBox ESRSAV Register
ISERR	R	IBox error status Register (from IBox)
IPASAV	R	IBox IPASAV Register
IVASAV	R	IBox IVASAV Register
MBDR	R	MBox Error Data Register
MSER	R	MBox Error address Register
MSTAT1	R	MBox status 1 Register
MSTAT2	R	MBox Status 2 Register
VIBASAV	R	IBox VIBASAV Register
VPCSITB	R	Valid PC Bits

Table 10-10 Supported Miscellaneous Register Names

Name	Access	Purpose
IBCPR	R	IBox CPR
PSL	R/D	Program Status Longword
SPAD#	R/D	Scratch Pad Address
STATE	R/D	STATE register
SVMOSAV	R	VMO save

EXAMINE EXAMINE [/space] [/next:hex_num] [/data_type] [{box_id}, reg_name]

"SPACE" can be any one of the following. The default is /PHYSICAL.

/SCRATCH access PBox scratch pad RAM space.

/GPRMEM access VAX processor GPR register space. See Table 10-7.

/FPUMEM access VAX processor FPU register space. See Table 10-8.

/PAMM access the PAMM RAMs.

/PHYSICAL the default, access VAX physical memory space.

/U access full PBM address space.

/VIRTUAL access VAX virtual memory space if the memory mapping is enabled, otherwise access physical address space and ignore address bits 30 and 31.

/data type can be any ONE of the following if the /space access is for /PHYSICAL, /VIRTUAL, or /U.

/BYTE access a single byte of data.

/LONG — the default, accesses a longword (4 bytes) of data.

/WORD — accesses a word (2 bytes) of data.

"Hex_adr" is the numeric address of the register or address being examined. Any of the following positional operators can be used to represent a "hex_adr" relative to the last address accessed. The default address is set to 0 by the INIT command.

^ Access last specified address

+ Access the address following the last specified address

- Access the address preceding the last specified address

@ Use the contents of the last specified address as the address to be accessed

"Reg_name" is a valid register name for GPRs (See Table 10-7), IPRs (See Table 10-8), Internal Registers (IR, See Table 10-9), or Miscellaneous registers (See Table 10-10).

"Hex_data" is the hexadecimal data to be deposited into the specified address space.

NOTE

To examine I/O space, the address and data must be preceded by a "%0" (percent Zero) to define the address/data as octal (instead of hexadecimal), the default.

XTND — XTND [/VAP, /NAPMOD=]

The default is XTND/NAP, and this command switches main memory, pointing at location 0, for a page aligned 64K block of good physical memory, as a Restart Parameter Block (RPB). It then is pointed to address plus 200 (hex) is loaded into the SP register.

HALT — Halt is used to halt the CPU by forcing it to the CSM halt loop.

The Macro PC is displayed and the CSM status code should be 11 (hex). If the CPU is already halted, a message to this effect will be printed, followed by the last recorded Macro PC and CSM status.

INITIALISE — INITIALISE [/switch]

"/switch" may be none, or any one of the following: /CPU, /EDURATION, /MICRO, or /FAVM. This command is used to initialize various facets of the CPU.

INIT

Without any switch, this command meets the requirements of DEC STD 032. The steps performed include:

- ACF2 register (enabled if 2Bus flag is on)
- BR, TR, and BIR registers are initialized
- ARBLVL is set to 4
- BISR, ICCS, MAPEN, and ZWS are set to 0
- NCCTL is set to 0 to enable MBox overlaps if the CPU clock is running full speed
- Turns off the TAX BINIs lamp
- The IBUFF and system cache are not affected

INIT/CPU

If the MBox has been loaded with system microcode, this command performs a full CPU initialization. It combines the CSH initialization with the steps for the INIT/KEY and INIT commands. The command does the following:

- Stop CPU clocks
- Master reset
- Loads and runs CSHV040 and CSHC041 overlays
- Sets MXTI flags ON
- Performs INIT/SCRATCH
- Performs INIT

INIT/SCRATCH

This command uses CSH to load the EBox scratch pad registers with the values needed to start the CPU. Data is taken from ECDDR.DPN, unless the filename is specified as INIT/SCR "filename.DPN". This file will initialize CPRs 80 - BF.

INIT/MICRO

This command will check KA8Bn.REV (n = 0 for 8600 and 5 for 8650) the microcode revision information. It will then execute ULOAD.[]M to load the system microcode according to the loaded microcode revision.

INIT/MAPN

This command determines the amount of physical memory and the number of IO adapters on the system and configures the DAMN accordingly. After an ABus INIT the IO adapter configuration registers are loaded with the number of megabytes of physical memory present, and SBI CYCLES IN and SBI CYCLES OUT are set in the SBIA CONTROL STATUS register.

LOAD

LOAD [/start:hex_adr] filename[.exe]

This command is used to load main memory with binary data taken from the specified file. If the /start: switch is used then the data is loaded starting at the specified "hex adr", otherwise, the data is loaded starting at location zero.

START

START [hex address]
This command is the same as a "EXAMINE/P" except that if a "hex address" is specified, it is used as the current PC at which to start the VAX programming. If the CPU is already running, then this command simply re-enters PTO mode without affecting the running CPU, even if "hex address" is included.

START/STEP

START/STEP [hex address]

This command prepares the CPU for micro-stepping macro instructions. It loads a CSM overlay, sets the PC if specified, and starts the CSM idle loop. The NEXT commands may then be issued to MTC or TNTC through the macro instructions.

When stepping is completed, the UNJAM command may be used to reset the CPUs to the start of the CSM idle loop without affecting the state of the machine.

NEXT

NEXT [hex_count]

This command single steps the VAX processor the specified number of macro instructions, or the default of one if no count is given. At the completion of the full step count, the PC of the next macro instruction is displayed and the command enters space-bar-step-mode, indicated by the >>> prompt at the end of the line instead of the beginning of the line.

UNJAM

UNJAM [IOA0, IOA1, IOA2, IOA3]
An UNJAM sequence is generated by the SBIA selected. SBIA or SBI may be used instead of IOA0 or IOA1.

VERIFY

VERIFY [/switch]

Switch is any of the following values: /ACCESS, /LOWMEM, /LEVEL, /ECC, /IBACK, /IBACK, /IDRAM, /ICS, /IDRAM, /MCF, /MCS, or /PAMM.

The contents of the selected CS/RAM is verified by comparing the data read from the CS/RAM with the contents of the .BIN file used to load it. If no switch is used, all the control stores and RAMs are verified.

This command will display the total number of discrepancies if the VAX debugger command set is not enabled. If VAX is enabled, each failure is reported with good/bad data.

10.4 THE HEX DEBUGGER

This section discusses the hardware debug and trace facility, or HEX, which can be enabled from either MACRO or DIAGNOSTIC contexts. The HEX commands allow the operator to modify and/or interrogate the state of the CPU.

The HEX command set is enabled by the "DEBUG" console command, and is indicated by the addition of one angle bracket (>) to the end of the current prompt. The MACRO prompt will become >>> and the DIAG prompt will become DC>.

10.4.1 The HEX Command Set

Table 10-1 contains a list of the HEX commands and a brief description of each.

Table 10-1 The HEX Command Set

CLEAR BREAK	CLEAR [BREAK, DBREAK] [Symbol, Reg, Hex_ID, All] "Symbol is a V$XARX visibility symbol name, "Reg" is a visibility register name, "Hex_ID" is a 16-bit hexadecimal ID number (see EXAMINE/CPU) and "All" will remove all breakpoints from the specified table. This command removes the specified breakpoint from the appropriate table. If "All" is specified, all of the breakpoints in that table are cleared. Example 10-7: CLEAR BREAK 1. >>>CLEAR BREAK V$x122 2. >>>CLEAR DBREAK ALL
CLEAR COUNT	This command clears the step counter. This counter keeps track of the number of machine steps taken or stepped since the last time the counter was cleared. This counter will also be cleared if you enter ESCAPE from EMIC or vice versa.
DEPOSIT [/CS]	DEPOSIT [/PREVIous:Hex_num] [/Switch] new_val Hex_adr "/Switch" can be any of the following: /ACCESS, /CONTROL, /CYCLE, /ECC, /EDRAM, /PBRAM, /PDRAM, /PCS, /IDRAM, /MCE, or /MCS "Hex_adr" can be a hexadecimal number or one of the following specifiers that determines the address relative to the last specified address. * Access last specified address - Access address following the last specified address - Access address preceding the last specified address

"Hex_data" is any hexadecimal number within the range limits of the specified control scope or RAM.

This command allows the operator to modify the contents of control store or RAM locations in the CAM ROYX CPU. The /NEXT switch can be used in combination with any other switch to perform successive deposits of the same data into consecutive locations.

NOTE

The MCR, ACCESS, and CYCLE RAMs can be deposited only if the system is out of the reset state. This will be true if the command "MICROSTEP 3" is executed.

Example 10-8: DEPOSIT

1. >>>>DEPOSIT/ACCESS 0 0 ;Deposit 0 into ACCESS RAM location 0

2. >>>>DEPOSIT/MCR/NEXT:10 0 0 ;Deposit 0 in the first 16 SBCE locations

3. >>>>DEPOSIT/CYCLE * OEA ;Deposit OEA into the assembly addressed CRX location

DEPOSIT/CHANNEL DEPOSIT/CHANNEL Hex_channel Hex_data

"Hex_channel" is one of the BBX channel numbers according to Table 10-12.

Table 10-12 BBX Control Channels

02 = VBA	09 = XBK
02 = PBN	0A = NCC
03 = IBB	0B = SBC
05 = ICA	0R = CSR
08 = SDP	10 = VBA

"Hex_data" is a 16-bit hexadecimal value to be deposited into the selected control channel.

This command deposits the specified data into the selected BBX control channel. Control channels vary in length, so if the data exceeds the length of the channel, the high order bits will be shifted out of the control channel and into the bit bucket. Table 10-13 lists the control channel bit positions.

Table 10-13 Control Channel Bit Positions

Channel	Bit	Signal Name
00 (VBA)	00	VBQ4 SBDADR 0 L
	01	VBQ4 SBDADR 1 L
	02	VBQ4 SBDADR 2 L
	03	VBQ4 SBDADR 3 L
	04	VBQ4 SBDADR 4 L

```
                05      MSG4 SELADR 5 L
                06      MSG4 SRDATR 6 D
                07      MSG4 SELADR 7 H.
                08      MSG4 SDDADR 0 L
                09      MSG4 SLBCTL 0 7

                10      MSG4 SDBCTL 1 L
                11      MSG4 SLBCTL 2 L

01 (XBN)        00      XNG2 SDP FDR A4 H
                01      XNG2 SDB FDR A5 H
                02      XNG2 SDB FDR A6 H
                03      XNG2 SDB FDR A7 H
                04      XNG2 SDB FDR A8 L
                05      XNG2 SDB FDR A9 L

07 (IBD)        00      IBDR SDB DA00 H
                01      IBDR SDB DA01 L
                02      IBDR SDB DA02 H
                03      IBDR SDB DA03 H
                04      IBDR SDB DA04 H

                05      IBDR SDB DA05 H
                06      IBDR SDB DA06 H
                07      IBDR SDB DA07 H
                08      IBDR SDB DA08 H
                09      IBDR SDB DA09 H

                10      IBDR SDB DA10 H
                11      IBDR SDB DA11 H
                12      IBDR SDB DA12 H
                13      IBDR SDB DA13 H
                14      IBDR SDB DA14 H
                15      IBDR SDB DA15 H

05 (ICA)        00      ICA1 ICS CNTRL CHNL 0
                01      ICA1 ICS CNTRL CHNL 1
                02      ICA1 ICS CNTRL CHNL 2
                03      ICA1 ICS CNTRL CHNL 3

0B (BDP)        00      BDPI DISA BYTE 32 PAR H
                01      BDPI DISA BYTE 10 PAR H
                02      BDPI FLIP MREG PAR H
                03      BDPI DISA SCC AP BDB H
                04      BDPI DISA ADL AR BDB H

                05      -BDPI FLIP GPRA H
                06      -BDPI FLIP GPRB H
                07      BDPI SPARE 0 H

09 (FSE)        00      ZBSH SDB CTL D00 H
                01      XBSH SDB CTL D01 H
                02      ZBSH SDB CTL D02 H
                03      XBXH SDB CTL D03 H
                04      ZBSH SDB CTL D04 H

                05      XBSH SDB CTL D05 H
                06      XBFH SDB CTL D06 H
                07      ZBEH SDB CTL D07 H
                08      ZBE2 SDB CTL D08 H
                09      XBLH SDB CTL D09 H
```

```
                0A (MCC)  00    MCCA LOAD XNA C
                          01    MCCA WRITE MICRO A L
                          02    Unused
                          03    MCCA SELFT_ENA H
                          04    MCCA WRITE M/CRO B C

                          05    Unused
                          06    MCCB DIAG INTR H
                          07    MCCB WRITE VIOL F
                          08    Unused
                          09    MCCB DIAG RAM WRITE H

                          10    MCCB CPR WRITE C
                          11    Unused

                0B (BBC)  00    BBC1 SDB CTL D00 E
                          01    BBC1 SDB CTL D01 E
                          02    BBC1 SDB CTL D02 H
                          03    BBC1 SDB CTL D03 E
                          04    BBC1 SDB CTL D04 H

                          05    BBC1 SDB CTL D05 D
                          06    BBC1 SDB CTL D06 E
                          07    BBC1 SDB CTL D07 C
                          08    BBC1 SDB CTL D08 E
                          09    BBC1 SDB CTL D09 C

                          10    BBC1 SDB CTL D10 C
                          11    BBC1 SDB CTL D11 E
                          12    BBC1 SDB CTL D12 D
                          13    BBC1 SDB CTL D13 H
                          14    BBC1 SDB CTL D14 D

                          15    BBC1 SDB CTL D15 H

                0E (CSB)  00    CSBR CKSL DBL FLAG F
                          01    CSBR CNSL DBL FLAG E
                          02    CSBR LOAD DIAG CNTR D
                          03    CSBR FLIP UNITS PAR H

                10 (VBA)  00    VB06 CLK CNTRL DATA 0 E
                          01    VB06 CLK CNTRL DATA 1 D
                          02    VB06 CLK CNTRL DATA 2 E
                          03    VB06 CLK CNTRL DATA 3 F
                          04    VB06 CLK CNTRL DATA 4 E

                          05    VB06 CLK CNTRL DATA 5 D
                          06    VB06 CLK CNTRL DATA 6 E
                          07    VB06 CLK CNTRL DATA 7 F
                          08    VB06 CLK CNTRL 8 B
                          09    VB06 CLK CNTRL 9 D

                          10    VB06 CLK CNTRL 10 D
                          11    VB06 CLK REG LD ENA B

DEPOSIT/CSPB    DEPOSIT/CSPB [/Switch] [/Mofile] Box_adr

                "Switch" can be any of the following:  /ECS,
                /FBACH, /FUNCS, /ICS, /PCH, /TORAM, or /TRAM
```

"Hex addr" is a hexadecimal address within the selected control store address range.

This command allows the operator to re-deposit to a control store location with bad parity. The data is taken from the .OBN file used to load the control store, and modified to contain bad parity (parity bit[s] are flipped). If the /Parity switch is specified, the .OBN file is not used, but the actual data in the RAM is used.

Correct parity can be restored by either reloading the entire control store (LOAD/CS for instance), or by using the DEPOSIT command with the original control store data. For the SBUS, UBUS, and MBUS, the DEPOSIT/MARK command can be used to restore the data and correct parity.

Example 10-9: DEPOSIT CSPE

1. >>>>DEPOSIT/CSPE/BCS 100

2. >>>>DEPOSIT/CSPE/BCS 07

DEPOSIT/MARK

DEPOSIT/MARK [/Switch] [/Sofile] Hex_addr [On, Off]

"Switch" can be either /BCS, /UCS, or /MCS

"Hexadr" is a hexadecimal address within the bounds of the selected control store.

This command allows the operator to set or clear the mark bit in the selected control store location. On/Off determines whether the mark bit is set or cleared. The data deposited to the control store location is taken from the .OBN file used to load the control store unless the /Sofile switch is used. Correct parity is maintained.

If the SMMM flag (Stop On Micro Mark) is set for the selected control store, the CPU clock will be stopped when a set mark bit is encountered.

When running at full system clock speed, the SMMM flag need not be cleared because the CPU clock is actually stopped beyond the selected micro-instruction. But, if the system clock is at 1/5 speed, the CPU clock will be stopped right at the selected micro-instruction. Attempts to TNIC or TSTATE will not succeed because the micro-instruction being executed has the mark bit set. Clearing the SMMM flag will allow the micro-instructions to be stepped.

Example 10-10: DEPOSIT MARK

1. >>>>DEPOSIT/MARK/BCS 100D ON

2. >>>>DEPOSIT/MARK/BCS 100D OFF

EXAMINE (CS) EXAMINE [/Execution_num] [/switch] hex_addr

"Switch" can be any of the following: /ACCESS, /CONTENT, /CPU#, /EBACS, /FBMCS, /EDMAN, /ICS, /IDMAN, /MCP, and /MCS.

"Hex addr" can be a hexadecimal number or relative to the last specified address as follows:

* Access last specified address.

+ Access address following last specified address. If "hex addr" is not specified, + is assumed.

- Access address preceding last specified address.

This command allows the operator to examine the contents of the selected control store at the specified address, or an address relative to the last specified address. The MCS, ACCESS, and CYCLE PANs can only be examined if the system is out of the TOOLS state. The command "MICROSTEP 1" can be used to ensure that this is true.

NOTE

The EXAM/MCS command performs a special function when the address is out of range of the MBox control store addresses (greater than FF). The command will read the state of the MCC shift path and convert it to .ULO format (MCDOS.ULO) and display the result. This allows the operator to inspect the data in the MCC shift path that has caused an MBox parity error.

Example 10-11: EXAMINE CONTROL STORE

```
1   >>>>EXAMINE/ECS/NEXT:10 0  ;Examine 10 (hex)
    ECS locations, starting at location 0

2.  >>>>EXAMINE/ICP   ;Examine ICP following last
    ECS address
```

EXAMINE/CHANNEL EXAMINE/CHANNEL [hex_channel]

"Hex channel" is the SDP channel number. See Table 10-14 for the SDP channel numbers.

Table 10-14 SDB Channel Numbers

00 - PBA	01 - PBM	02 - MCD
03 - IBC	04 - IDP	05 - ICA
06 - ICM	07 - CLK	08 - EDP
09 - SBS	0A - MCC	0B - MAP
0C - KSD	0D - SBC	0E - CSB
0F - CSA	10 - ICA0	11 - TCA1
12 - TD42	13 - ICA3	14 - RTM
15 - RESERVED	16 - RESERVED	17 - RESERVED

NOTE

The RTM visibility channel would have a "R" in the VS symbol, as VSR123, whereas channels 00 through 0F would have 0 - F.

This command will display all visibility bits in the specified SDB channel. The data is displayed as a string of hexadecimal characters followed by a three character checksum. The number of characters displayed is:

PBA/PBM — 72 characters. The PBMs shift channels are 12 bits in length. There are 24 shifts which provides 288 bits.

IDP/IED/TCD — 64 characters. The IBx shift paths are 8 bits in length, and there are 32 shifts which provides 256 bits.

All others — 48 characters. The rest of the shift paths are also 8 bits. With 24 shifts each, this provides 192 bits.

Example 10-12: EXAMINE/CHANNEL

1. >>>EXAMINE/CHAN 0

2. >>>EXAMINE/CHAN 14

EXAMINE/SDB — EXAMINE/SDB [symbol_name], [Register_name], SDB_ID, "Signal_name"

"Symbol_name" is the VS symbol assigned to the particular visibility signal by the SDB CAD (.CDF) files.

"Register_name" is the name of a default visibility register or a visibility register defined within SEX by the TRACE DEFINE command. A visibility register is a group of VTRAN bits joined together in a logical arrangement. Table 10-15 contains the names of the default visibility registers. See also the TRACE DEFINE command. A list of the default visibility registers wouldn't be complete without the contents of each of the registers. See Table 10-16. Table 10-16 will be located at the end of this section due to its length.

Table 10-15 Names of Default Visibility Registers

ABUS	ARA H	ARBUS	ALYSLS
BBUS	BBELSS	BDPPB	DPORE
BNCF	BSTALL	CIPL	EVABLS
FASHS	FALPC	FMUPC	IRDBUF
IDCE	IRXSRF	IRIAG	IXFH
LOPASL	IUPC	IVADUC	MSFUCI
MOBUSH	MSNREG	MUPC	NAIRAM
OPAH	OPEJS	OPCODB	OPLCP
OPPORT	PAACK	PAMD	PARH
PAXITY	PSG	RECBUS	STALL
UPCSAV	WDCO		

"Hex_ID" is a 16-bit SDB or register ID. SDB ID's are hardware visibility-bit addresses and are defined in the .CDP files. REG_IDs are software-defined numerical tags for visibility registers.

"Signal name" is the full visibility signal name, including the High/Low specified.

NOTE

The signal_name must be enclosed in quotes ("signal name"). Also, the .CDP files do not contain any LOW signal names (i.e., the signal "THIS SIGNAL L" would be listed as "-THIS SIGNAL H".

This command displays the name and state of a single visibility signal, or the name and state of a visibility register.

Example 10-13: EXAMINE/SDB

```
1.  >>>>EXAMINE/SDB V$A123          ;Get symbol name

2.  >>>>EXAMINE/SDB W3U3            ;New register name

3.  >>>>EXAMINE/SDB 1               ;View SDB ID

4.  >>>>EXAMINE/SDB #00B            ;View register ID
```

EXIT This command disables the SBX command set.

MICROSTEP MICROSTEP [Hex_number]

"Hex_number" is a hexadecimal wrap count in the range of 1 to 10C. The default is 1.

The MICROSTEP command causes "Hex_number" of microinstructions to be executed at full system clock speed. Before stepping begins, the CPU clock is forced to T3 state. CPU clocks must be stopped for this command to function.

The execution of a MICROSTEP command invokes space-bar-step-mode (SBSM), which is indicated by the SYSN>> or USRMOD>> prompts. In space-bar-step-mode, the next step can be executed by depressing the spacebar.

SEQX contains an error until any key other than the spacebar is depressed.

After sampling has finished, all GPIO's are displayed.

REPORT

The REPORT command will display all default trace visibility register data. REPORT is invoked on completion of any TXICRO or TSTATX.

The trace reported assumes that all visibility registers are 48 bits in length. If a register greater than 48 bits in length is added to the trace list, the display will be truncated to 48 bits. A "D" will precede the register in the trace list to indicate the truncation.

Any register or signal value whose value is different from that observed on the previous report will be prefixed with an asterisk (*) to denote the change.

SET BREAK

SET [ABREAK, DBREAK] [Symbol, Reg, Hex_ID] [Span, Value:val]

"Symbol" is a VSXXXX visibility symbol name assigned from the .CDF file.

"Reg" is the name of a visibility register defined within HSX or by the TRACK DEFINE command.

"Hex ID" is a 16-bit DDR or register ID. DDR_IDs are hardware visibility-bit addresses and are defined in the .CDF files. REG_IDs are software defined for visibility registers.

"Span" and "Value:val" are optional breakpoint parameters.

Up to 4 breakpoints can be set in each table (DBREAK, ABREAK). If the signal or register specified is already in the table, the user will be asked if he wants to alter the parameters of that table entry. If so, new parameters are accepted.

The TXICRO or TSTATX command will stop and report the current machine state only when a break condition is met, when all of the "AND" conditions are true, or any of the "OR" conditions are true.

The "SPAN" option causes a break condition when that signal or register is in a state other than what it was when the trace was started.

The "Value:val" option causes a break condition when that signal or register equals the value specified (1 or 0).

If neither option is selected, "Change" is implied, and a break will occur every time the selected signal or register changes.

Example 10-14. SET BREAK

```
1.  >>>>SET ABREAK V5125          ;break when  V5125
    changes

2.  >>>>SET ABREAK V5456 VAL:1    ;Break when  V5456
    equals a 1

3.  >>>>SET OBREAK ABUS NOAM      ;Break if ABus is
    different than it is now
```

SET HISTORY — For use by engineering during the development of microcode.

SET MARGIN — SET MARGIN [High, Low, Normal] [1A, B, C, DE, FG, All]

This command allows the operator to adjust the output level of the power supplies for purposes of detecting marginal faults. Margins can be set +5% (High) or -5% (Low). The outputs of the D and E regulators and the F and G regulators are tied together so these regulators must be margined together (DE, FG).

The regulator outputs to be margined are specified by the second argument (A, B, C, DE, FG). If no regulator is specified, the default is "All".

SHOW BREAK — This command shows the contents of the ABREAK or OBREAK tables. The signal or register ID, name, and options) break conditions are shown for each table entry.

SHOW DEFINE — SHOW DEFINE [Reg_name, Hex_ID]

"Reg_name" is the name of a visibility register defined within ICE or by the TRACE DEFINE command. See Table 10-15 for a list of the default visibility registers.

"Hex_ID" is a 16-bit SDB or Register ID. SDB IDs are hardware visibility-bit addresses and are defined in the .CDF files.

REG IDs are software defined numerical tags for visibility registers.

This command displays a list of all visibility symbol names (V$NNNN) used to construct the visibility register. The display is organized in descending order (high order to low order, left to right with 12 (decimal) terms per line. The last four characters of each V$NNNN (NNNN) symbol is shown.

Example 10-15: SHOW DEFINE

```
1.  >>>>SHOW DEFINE REGNAM

    NAME  REGNUM  ID.     VSNAME. Name, and number
                          of VTERMs that make up
                          this register
    1001 0000 0000 0004
    0005 0006 0007 0008   The symbols for the ID
    0009 0010 0011 0012   vtnums in this
    0013                  register (VSNAME)
```

SHOW HISTORY

```
SHOW  HISTORY  [/Output="filename[.DAT]]  [Hex_num]
[Hex_num]
```

The behavior of this command is unpredictable if the optional history module is not in place in the system. It is designed to be used by engineers during the development of microcode.

SHOW NAME

```
SHOW  NAME  [Symbol_name,  Reg_name,  Hex_ID,
"Signal_name"]
```

"Symbol_name" is the name of a visibility register defined within SEX or by the IRACE DEFINE command. See Table 10-10 on Page 98 for a list of default visibility registers with bit definitions.

"Hex ID" is a 16-bit REG or Register ID. REGs are SEX hardware visibility-bit addresses and are defined in the .COF files. Reg IDs are software defined numerical tags for visibility registers.

"Signal_name" is the full visibility signal name, including the "HIGH" or "LOW" specified. The signal name must be enclosed in quotes ("signal name").

This command looks up the symbol name, register name, ID, or signal name and prints the corresponding VS symbol, signal/register name and ID. If displaying a register, the size (number of vtnums) is also displayed.

Example 10-16: SHOW NAME

```
1.  >>>>SHOW NAME VSSL53                 Symbol name
    5145  VSSL63  IOD DRAM LAST H

2.  >>>>SHOW NAME STALL                  Register name
    8020  STALL  29.

3.  >>>>SHOW NAME 8000                   Register ID
    8000 LOPC  15.

4.  >>>>SHOW NAME "IOD DRAM LAST H"      Signal name
    5145  VSSL63 IOD DRAM LAST H
```

SHOW REGISTER

This command displays a list of all visibility registers currently defined, whether within SCX or by the TRACE DEFINE command. The display includes each register's ID number, name, and size.

NOTE

A validity check is done on each register before it is displayed. If a bad entry in a register is found, the name "=badreg-*" is assigned to the register. Because of the bad register definition, there is no way for the code to display the true name of the register.

SHOW TRACE

This command displays a list of all registers and signals currently selected for tracing. The trace list may be changed by the user using the TRACE ADD command. The current breakpoint options are also included.

STATESTEP

STATESTEP [Hex_num]

"Hex_num" is a hexadecimal step count in the range of 1 to 400 (max). The default is 1 step.

The STATESTEP command causes "Hex_num" clock ticks to be executed.

A STATESTEP count of 4 is equivalent to one MICROSTEP. The micro-pc's of the boxes are not shown for STATESTEP. Statesteps are executed at full machine speed.

The execution of the STATESTEP command invokes space-bar-step-mode (SBSM) as indicated by the SBSM>> or SBSMCC>> prompts. In SBSM, the user can cause additional steps to be executed by depressing the space-bar (one STATESTEP per space character). SBSM remains in effect until a character other than the space-bar is depressed.

THICKO, TSTATE

THICKO [Hex_num], TSTATE [Hex_num]

"Hex_num" is a hexadecimal step count. If not specified, the default is 1.

These commands initialize the trace facility and execute Ctrl clock steps (1 clock cycle for THICKO (THIC), 1 clock phase for TSTATE) while watching for a trace breakpoint condition, or until the step count expires. The state of the machine is then captured and reported in accordance with the current trace settings (see SHOW TRACE).

If any trace breakpoints are set, trace stepping stops, and the trace report generated, as soon as a snake condition is met. If no step count is given, and trace breakpoint conditions are defined, then stepping continues indefinitely until either a breakpoint or Ctrl right character is detected.

When a breakpoint is detected, or the step count expires, XBX enters XBDM.

NOTE

If DOIBT is ON, no trace information is displayed until the final trace step has completed; if DOIBT is OFF, then trace information is displayed for each step.

TNTC is equivalent to MICROSTEP followed by XBXCont. DSTATE is equivalent to STATESTEP followed by report.

TRACE ADD

TRACE ADD item

Item is any combination of "Register" (add registers to the trace list) or VSNEXP (add signals to the trace list).

This command is used to customize the register and signal trace lists (see SHOW TRACE). The list of items to add is included in the command line and may specify as many items as will fit on the remainder of that line. The %$ prefix is required. Use a SHOW REGISTER command to display available registers.

Example 10-17: TRACE ADD

```
1. >>>TRACE ADD IDPC USERSO PARITY V$19C1 V$9345
```

TRACE DEFINE

TRACE DEFINE Reg_name

"Reg_name" is a unique name to be assigned to the new "user-defined" register. Name strings must be 6 characters or less in length, and will be truncated to 6 characters if names of more than 6 characters are entered.

This command is used to define visibility registers in addition to those already defined (See Table 10-15). The command will check to see if the new name is not already defined and if not, will prompt the user to enter a list of VS symbol names. Only the last four characters of the six character VSNNRM [NNNN] need be entered. Enter the signals beginning with the most significant. Signals must be separated with a space or a comma, but not both.

Signal names may be continued on the next line with a CR (Carriage Return) at the end of the line. Terminate the definition with an additional carriage return. XBX will display the verification. Note in the example NNN, the register number, NAMEREG, the register name (truncated to 6 characters), and n, the number of signals in the register. A register defined with TRACE DEFINE is added to the default trace list (SHOW TRACE).

Example 10-18: TRACE DEFINE

```
1.  >>>TRACE DEFINE newregister
    Enter VS names separated with <SP> or <,>
    0001 0002 0003 0004 0005 0006 0007 <CONTROL>
    0009 0010 0011 0012 0013 0014 0015 0016<CR>
    <CR>
    ENTER NEWREG n.
```

TRACE DELETE — TRACE DELETE Reg_name

This command will delete a user defined visibility register which was created by TRACE DEFINE. The command will ask for confirmation before taking any action. If the delete request is confirmed, the named register is removed from the register list and removed from the register trace list (TRACE REMOVE). Registers in the default register list may not be deleted.

Example 10-19: TRACE DELETE

```
1.  >>>TRACE DELETE newreg
    Delete register NEWREG -- confirm (Y/N)  y
    (Caution response)
```

TRACE REMOVE — TRACE REMOVE Item

"Item" can be any of the following:

R*	Remove all registers from the list
S*	Remove all signals from the list
Reg name	Remove named registers from the list
VSNNN	Remove named signal from the list

This command is used to customize the register and signal trace list by allowing unwanted information to be removed. The list of items to remove is included in the command line and may specify as many items (separated by a space) as will fit on the remainder of the line. The VS is used for signal names. A default register will just be removed from the trace list, it will not be removed from the register list.

Example 10-20: TRACE REMOVE

```
1.  >>>TRACE REMOVE VS1234 VS4567 NEWREG
```

TRACE RESTORE — This command is used to restore the default trace list (including all default visibility registers and signals) and zero all counters and variables.

10.4.2 Default Visibility Registers

The HSX debugger command set provides access to a number of pre-defined visibility registers useful for tracing the machine state. Most of these registers are part of the default ISAVE list used by the SHOW, ISTATE, and SETPT commands. The SHOW ADD command can be used to add a register to the ISAVE list. The ISAVE DEFINE command can be used to define a new visibility register. Table 10-15 lists the visibility registers present in the console program.

Just the name of the register is not enough when trying to decipher a VBIC, or any trace; a bit breakdown of the registers is necessary. Table 10-16 is a complete list of visibility registers and the signal names that make them up. Note that bits are listed in order of the most-significant bit first. The special filler symbol 'XXXX' is used to 0-pad a visibility register in places where signals are no longer defined or where nibble alignment is desired. Visibility register definition can be found in the source listing for the HSXBOX module (HSXDBXTI.CMT).

Table 10-16 Default Visibility Registers

ABUS

VP symbol	Bit	Nibble	Signal name
V$A223	43	11	ST ABUS IOA REQUEST 3 H
V$A22x	42		SB ABUS IOA REQUEST 2 H
V$A215	41		ST ABUS IOA REQUEST 1 H
V$A217	40		SB ABUS IOA REQUEST 0 H
V$A100	39	10	ABUS CPU SUP DONE H
V$A226	38		ABUS CPU RDY XMASK H
V$A210	37		ABUS LEM STAT 1 H
V$A222	36		ABUS LEM STAT 0 H
V$A211	35	9	ABUS CMD MASK 3 H
V$A2.4	34		ABUS CMD MASK 2 H
V$A219	33		ABUS CMD MASK 1 H
V$A210	32		ABUS CMD MASK 0 H
V$B125	31	8	ABUS DATA ADDR8 31 H
V$B203	30		ABUS DATA ADDR8 30 H
V$B139	29		ABUS DATA ADDR8 29 H
V$b128	28		ABUS DATA ADDR8 28 H
V$B100	27	7	ABUS DATA ADDR8 27 H
V$B197	26		ABUS DATA ADDR8 26 H
V$B198	25		ABUS DATA ADDR8 25 H
V$B103	24		ABUS DATA ADDR8 24 H
V$B200	23	6	ABUS DATA ADDR8 23 H
V$b130	22		ABUS DATA ADDR8 22 H
V$B201	21		ABUS DATA ADDR8 21 H
V$b131	20		ABUS DATA ADDR8 20 H
V$B189	19	5	ABUS DATA ADDR8 19 H
V$B135	18		ABUS DATA ADDR8 18 H
V$B136	17		ABUS DATA ADDR8 17 H
V$B146	16		ABUS DATA ADDR8 16 H

VS symbol	Bit	Nibble	Signal name
V$S140	15	4	ABUS DATA ADDR 15 P
V$B190	14		ABUS DATA ADDR 14 P
V$N141	13		ABUS DATA ADDR 13 N
V$B191	12		ABUS DATA ADDR 12 S
V$B137	11	3	ABUS DATA ADDR 11 M
V$A196	10		ABUS DATA ADDR 10 K
V$B208	09		ABUS DATA ADDR 09 K
V$B199	08		ABUS DATA ADDR 08 H
V$B194	07	2	ABUS DATA ADDR 07 M
V$B169	06		ABUS DATA ADDR 06 H
V$B209	05		ABUS DATA ADDR 05 H
V$B206	04		ABUS DATA ADDR 04 H
V$B205	03	1	ABUS DATA ADDR 03 M
V$B165	02		ABUS DATA ADDR 02 D
V$B176	01		ABUS DATA ADDR 01 M
V$B177	00		ABUS DATA ADDR 00 D

ABADR

V$ symbol	Bit	Nibble	Signal name
V$K103	28	8	MAP1 ARRAY ADR 28 M
V$K105	27	7	MAP1 ARRAY ADR 27 M
V$K101	26		MAP1 ARRAY ADR 26 H
V$K170	25		MAP1 ARRAY ADR 25 D
V$K174	24		MAP1 ARRAY ADR 24 M
V$K177	23	6	MAP1 ARRAY ADR 23 M
V$K173	22		MAP1 ARRAY ADR 22 M
V$K171	21		MAP1 ARRAY ADR 21 D
V$K159	20		MAP1 ARRAY ADR 20 M
V$K162	19	5	MAP1 ARRAY ADR 19 D
V$K161	18		MAP1 ARRAY ADR 18 M
V$K157	17		MAP1 ARRAY ADR 17 M
V$K156	16		MAP1 ARRAY ADR 16 D
V$K155	15	4	MAP2 ARRAY ADR 15 H
V$K151	14		MAP2 ARRAY ADR 14 M
V$K154	13		MAP2 ARRAY ADR 13 M
V$K146	12		MAP2 ARRAY ADR 12 H
V$K140	11	3	MAP2 ARRAY ADR 11 D
V$K143	10		MAP2 ARRAY ADR 10 M
V$K146	09		MAP2 ARRAY ADR 09 D
V$K142	08		MAP2 ARRAY ADR 08 M
V$K145	07	2	MAP2 ARRAY ADR 07 M
V$K141	06		MAP2 ARRAY ADR 06 D
V$K140	05		MAP2 ARRAY ADR 05 M
V$K139	04		MAP2 ARRAY ADR 04 M
V$K138	03	1	MAP2 ARRAY ADR 03 D
V$XXX	02		Zero-Filler
V$XXXX	01		Zero-Filler
V$XXXX	00		Zero-Filler

ARBUS

VS symbol	Bit	Nibble	Signal name
V$K113	31	8	ARRAY BUS D31 H
V$K112	30		ARRAY BUS D30 H
V$K106	29		ARRAY BUS D29 H
V$K107	28		ARRAY BUS D28 H
V$K105	27	7	ARRAY BUS D27 H
V$K102	26		ARRAY BUS D26 H
V$K175	25		ARRAY BUS D25 H
V$K178	24		ARRAY BUS D24 H
V$K104	23	6	ARRAY BUS D23 H
V$K100	22		ARRAY BUS D22 H
V$K176	21		ARRAY BUS D21 H
V$K172	20		ARRAY BUS D20 H
V$K167	19	5	ARRAY BUS D19 H
V$K169	18		ARRAY BUS D18 H
V$K166	17		ARRAY BUS D17 H
V$K174	16		ARRAY BUS D16 H
V$K131	15	4	ARRAY BUS D15 H
V$K127	14		ARRAY BUS D14 H
V$K129	13		ARRAY BUS D13 H
V$K130	12		ARRAY BUS D12 H
V$K126	11	3	ARRAY BUS D11 H
V$K125	10		ARRAY BUS D10 H
V$K128	09		ARRAY BUS D09 H
V$K119	08		ARRAY BUS D08 H
V$K123	07	2	ARRAY BUS D07 H
V$K124	06		ARRAY BUS D06 H
V$K122	05		ARRAY BUS D05 H
V$K118	04		ARRAY BUS D04 H
V$K121	03	1	ARRAY BUS D03 H
V$K117	02		ARRAY BUS D02 H
V$K120	01		ARRAY BUS D01 H
V$K116	00		ARRAY BUS D00 H

DBUS

VS symbol	Bit	Nibble	Signal name
V$D271	31	8	-DBUS D31 H
V$D267	30		-DBUS D30 H
V$D272	29		-DBUS D29 H
V$D265	28		-DBUS D28 H
V$D273	27	7	-DBUS D27 H
V$D225	26		-DBUS D26 H
V$D2[illegible]	25		-DBUS D25 H
V$D240	24		-DBUS D24 H
V$D270	23	6	-DBUS D23 H
V$D269	22		-DBUS D22 H
V$D274	21		-DBUS D21 H
V$D275	20		-DBUS D20 H

V$ symbol	Bit	Nibble	Signal name
V$3171	19	5	-DBUS D19 H
V$D107	18		-DBUS D18 H
V$3237	17		-DBUS D17 H
V$7216	16		-DBUS D16 H
V$J269	15	4	-DBUS D15 H
V$3268	14		-DBUS D14 H
V$J274	13		-DBUS D13 H
V$3267	12		-DBUS D12 H
V$3220	11	3	-DBUS D11 H
V$J242	10		-DBUS D10 H
V$3230	09		-DBUS D09 H
V$J247	08		-DBUS D08 H
V$3176	07	2	-DBUS D07 H
V$3178	06		-DBUS D06 H
V$3105	05		-DBUS D05 H
V$3104	04		-DBUS D04 H
V$3149	03	1	-DBUS D03 H
V$D164	02		-DBUS D02 H
V$3243	01		-DBUS D01 H
V$3126	00		-DBUS D00 H

8uPISB

V$ symbol	Bit	Nibble	Signal name
V$3198	04	2	-ICA BCF PLCBK KBOS) L
V$3230	03	1	-ICA6 BPLPB PE CPC LAT H
V$X218	02		-CPU BPCP 2 A H
V$3242	01		ICA6 IPCBK CTL 2 H
V$3194	00		-CPU BPIP 0 A H

EDPPE

V$ symbol	Bit	Nibble	Signal name
V$D134	43	11	EDP EDPE D3 H
V$D137	42		EDP EDPE D2 H
V$4102	41		EDP EDPE D1 H
V$D136	40		EDP EDPE D0 H
V$E138	39	10	EDP STATE 7 H
V$E109	38		EDP STATE 6 H
V$E177	37		EDP STATE 5 H
V$E108	36		EDP STATE 4 H
V$E184	35	9	EDP STATE 3 H
V$E136	34		EDP STATE 2 H
V$E110	33		EDP STATE 1 H
V$E179	32		EDP STATE 0 H
V$A190	31	8	ICA ISTALL A H
V$9193	30		EDP PEP MODE H
V$9190	29		ICA BLOC PE H
V$9191	28		ICA IEUC PE H

VS symbol	Bit	Nibble	Signal name
V$5123	27	7	TCA) TCS PAR XBR H
V$9101	26		ICP IDPAN PS B
V$9176	25		IDP TANDX PX H
V$9142	24		IDP IBMUX PS H
V$XXXX	23	6	Zero-Fillers
V$6127	22		MCC OP PA ACK E H
V$9127	21		MCC IBP PA ACK B
V$9162	20		MCC MBUX CN P2 H
V$3221	19	5	XBR XBYCAL TO ICA H
V$9106	18		SBS IBUX SBX LTH A H
V$9170	17		XBR ICS PC FLAG H
V$9110	16		SBD SCB PS LSV CYC H
V$D166	15	4	XBR XBUX XBR LST CYC H
V$4167	14		SBD SBOX SBR TO IDP H
V$9170	13		XBR XCS PC FLAG H
V$9113	12		SBD SCB PS LSV CYC B
V$D156	11	3	SBD SDP PE FLAG A H
V$9169	10		SBD XDP PF FLAG H
V$9173	09		SBD XBUX PK FLAG H
V$9161	08		XCH RX PARAP H
V$9176	07	2	XBH IBUX PE FLAG H
V$9145	06		SBD MBUS PB FLAG B
V$C159	05		-EXC XCC RAM P4R XBR H
V$C141	04		BBD6 EVC D0P E
V$4139	03	1	-ICA BMCX CBK XIIE CIX B
V$6131	02		XBX MBUS DPAR P2 H H
V$6129	01		SBS MBUS DPAR P1 C H
V$4281	00		-DMAR DD6 H

BPCXRK

V$ Symbol	Bit	Nibble	Signal name
V$6142	31	3	-CSH EBOX FORK E B
V$8180	07	2	-CSH EBOX FORK D H
V$8124	06		-CSH EBOX FORK C B
V$5159	05		-CSH EBOX FORK B H
V$C226	04		-CSH EBOX FORK A B
V$XXXX	03	1	Zero-Fillers
V$C102	02		-CSH EBOX IRD A H
V$6101	01		-CEB BBCX IRD B H
V$1231	00		-CSH EBOX IRD C H

BMCF

VS symbol	bit	Nibble	Signal name
V$D161	39	10	CSA BMCF 5 A H
V$D166	38		CSA BMCF 4 A H
V$D181	35	9	CSA BMCF 3 A H
V$D18J	34		CSB BMCF 2 A H
V$J180	33		CSb BMCF 1 A H
V$D182	32		CSB BMCF 0 A H
V$XXXX	31	8	Zero-Filler
V$XXXX	30		Zero-Filler
V$A153	29		BBC BBOX MCP 5 H
V$A182	28		BBC BBOX MCP 4 H
V$A174	27	7	BBC BBOX MCP 3 H
V$A165	26		BBC BBOX MCP 2 H
V$A16C	25		BBC BBOX MCP 1 H
V$A154	24		BBC BBOX MCP 0 H
V$XXXX	23	6	Zero-Filler
V$D130	22		-BBC9 MCP-LOAD K H
V$D154	21		-BBC9 MCP-LOAD I H
V$D150	20		-BBC9 MCP-BROWN N H
V$D151	19	5	-BBC9 MCP-RFQUK T H
V$D146	18		-BBCA MCP-EB MBOX E
V$D140	17		BBCA MCP-MCMK H
V$D150	16		BBCK MCP-E DISP I H
V$C179	15	4	-BBC MCP-ACK REQ LTH H
V$C187	14		-BBC MCP-CLR BBFS WTH H
V$C184	13		-BBC MCP-CLR IBDS LTE H
V$C186	12		-BBC MCP-CLR OPFS LTH H
V$C180	11	3	-BBC MCP-RK TRNSTL LTH H
V$C178	10		-BBC MCP-MEM RBD LTH H
V$C127	09		-BBC MCP-MEM XRT LTH H
V$C199	08		-BBC MCP-CP MBC LTH H
V$C141	07	2	-BBCA MCP-ACK RBQ H
V$C138	06		-BBCB MCP-CLR BBFS H
V$C141	05		-BBCB MCP-CLR IBDS H
V$D137	04		-BBCB MCP-CLR OPFS H
V$D134	03	1	-BBC7 MCP-BR OMSTL H
V$D139	02		-FBC7 MCP-MEM RBQ H
V$D104	01		-LBC7 MCP-MBA MBC H
V$D.05	00		-CBC7 MCP-CP MBT H

BSTALL

V$ symbol	Bit	Nibble	Signal name
V$E103	11	7	-EBD BSTALL TO CBS H
V$C[illegible]	10		SBD BSTALL TO ICC H
V$D165	09		BBD BSTALL TO EBD H
V$D160	08		RBD RSTALL TO FRP H
V$D190	07	2	[illegible] TO FRA L
V$1123	06		BBD BSTALL TO FBM H
V$J157	05		-BBD RSTALL TO IRI H
V$5221	04		JBJ JSTALL TO ICA H
V$6138	03	1	BBD BSTALL TO ICH H
V$4136	02		XBJ JSTALL TO IRF H
V$A184	01		BBD BSTALL TO NCL H
V$X114	00		[illegible] TO MCD H

BUSC

V$ symbol	Bit	Nibble	Signal name
V$P120	12	4	CSB CPC 12 H
V$P133	11	3	CSB CPC 11 H
V$P111	10		CBB CBC 10 H
V$P110	09		CSB CPC 09 H
V$P107	08		CBB CBC 08 H
V$P112	07	2	CSB CBC 07 H
V$P[illegible]	06		CSB CPC 06 H
V$E109	05		CSB CBC 05 H
V$P[illegible]0	04		CSB CPC 04 H
V$[illegible]4	03	1	CSB CPC 03 H
V$E126	02		CSB CPC 02 A H
V$P[illegible]8	01		CSB CBC 01 A H
V$E127	00		CSB CBC 00 A H

EVADDR

V$ symbol	Bit	Nibble	Signal name
V$H2[illegible]2	31	8	EDP EVA A31 H
V$H112	30		BDE EVA A30 H
V$H107	29		EDP EVA A29 H
V$H181	28		BDE EVA A28 H
V$H188	27	7	EDP EVA A27 H
V$H132	26		BDE EVA A26 H
V$H201	25		XDP EVA A25 H
V$J119	24		JDP EVA A24 H
V$J113	23	6	BDC EVA A23 H
V$J181	22		XDP EVA A22 H
V$K134	21		XDP EVA A21 H
V$K129	20		BDP EVA A20 H

VS symbol	Bit	Nibble	Signal name
V$0126	19	5	EDP BVA A19 H
V$0127	18		EDP BVA A18 H
V$0122	17		EDP BVA A17 H
V$0207	16		EDP BVA A16 H
V$0205	15	4	EDP BVA A15 H
V$0143	14		EDP BVA A14 H
V$0146	13		EDP BVA A13 H
V$0142	12		EDP BVA A12 H
V$0144	11	3	EDP BVA A11 H
V$0214	10		EDP BVA A10 H
V$0213	09		EDP BVA A09 H
V$0175	08		EDP BVA A08 H
V$0171	07	2	EDP BVA A07 H
V$0170	06		EDP BVA A06 H
V$0176	05		EDP BVA A05 H
V$0174	04		EDP BVA A04 H
V$0167	03	1	EDP BVA A03 H
V$0170	02		EDP BVA A02 H
V$0159	01		EDP BVA A01 A H
V$0162	00		EDP BVA A00 A H

VA0U3

VS symbol	Bit	Nibble	Signal name
V$1197	31	8	-PDA EA BUS D31 H
V$1209	30		-PDA EA BUS D30 H
V$1135	29		-PDA EA BUS D29 H
V$1189	28		-PDA EA BUS D28 H
V$1171	27	7	-PBA PA BUS D27 H
V$1167	26		-PBA PA BUS D26 H
V$1250	25		-PBA PA BUS D25 H
V$1170	24		-PBA PA BUS D24 H
V$1194	23	6	-PBA PA BUS D23 H
V$1240	22		-PBA PA BUS D22 H
V$1149	21		-PBA PA BUS D21 H
V$1165	20		-PBA PA BUS D20 H
V$1251	19	5	-PBA PA BUS D19 H
V$1248	18		-PBA PA BUS D18 H
V$1137	17		-PBA PA BUS D17 H
V$1169	16		-PBA PA BUS D16 H
V$1141	15	4	-PBA PA BUS D15 H
V$1252	14		-PBA PA BUS D14 H
V$1134	13		-PBA PA BUS D13 H
V$1152	12		-PBA PA BUS D12 H
V$1263	11	3	-PBA PA BUS D11 H
V$1244	10		-PBA PA BUS D10 H
V$1246	09		-PBA PA BUS D09 H
V$1164	08		-PBA PA BUS D08 H

V8 symbol	Bit	Nibble	Signal name
V$1200	07	2	-EBA EA BUS D07 F
V$1205	06		-EBA EA BUS D06 E
V$1104	05		-EBA EA BUS D05 D
V$1109	04		-EBA EA BUS D04 R
V$1264	03	1	-EBA EA BUS D03 N
V$1265	02		-EBA EA BUS D02 B
V$1251	01		-EBA EA BUS D01 M
V$1166	00		-EBA EA BUS D00 T

EAOPC

V8 symbol	Bit	Nibble	Signal name
V$0243	08	2	EA[CPCA A8 R
V$0321	07	2	EA11 CPCA A7 M
V$0320	06		EA11 CPCA A6 L
V$0319	05		EA11 CPCA A5 K
V$0324	04		EA11 CPCA A4 E
V$0424	03	1	EA11 CPCA A3 F
V$0322	02		EA11 CPCA A2 C
V$0247	01		EA11 CPCA A1 B
V$0335	00		EA11 CPCA A0 H

PNCPC:

V8 symbol	Bit	Nibble	Signal name
V$1102	08	2	PN1[CPC A8 K
V$1149	07	4	PN11 CPC A7 N
V$1179	06		PN11 CPC A6 H
V$1177	05		PN11 CPC A5 H
V$1155	04		PN11 CPC A4 H
V$1154	03	1	PN11 CPC A3 T
V$1160	02		PN11 CPC A2 B
V$1153	01		PN11 CPC A1 D
V$1150	00		PN11 CPC A0 E

IBDBLK

V8 symbol	Bit	Nibble	Signal name
V$4289	42	11	IBD DRAM CTX 2 D
V$4286	41		IBD DRAM CTX 1 W
V$4287	40		IBD DRAM CTX 0 B
V$5117	39	10	IBD DRAM TYPE 1 U
V$5211	38		IBD DRAM TYPE 0 H
V$5116	37		IBD DRAM REF 1 K
V$5214	36		IBD DRAM REF 0 H

VS symbol	Bit	Nibble	Signal name
V$5102	35	8	IBD SELECT NEXT B
V$5163	34		IBD SCAN CAST H
V$5173	33		IBD DRAM BUSY H
V$5236	32		IBD DISPOLY SO ??? H
V$5205	31	7	IBD BUF FJ INST H
V$5249	30		IBD SXI CPC H
V$6179	29		IBD BUF IBUF PT I
V$6128	28		IBD TOV DRAM PE E
V$5270	27	7	IBD IFCHK VALID H
V$5125	26		IBD DMUX VALID H
V$5172	25		IBD RAC ONLY H
V$5213	24		IBD BPC 0 OR 7 H
V$5111	23	6	IBD ISEL B2 C
V$3193	22		IBD LD IFCHK DISP H
V$5101	21		IBD BUF LD CTL I H
V$5103	20		IBD BUF LD CTL C E
V$4286	19	5	IBD BUF DELTA PC 3 H
V$4197	18		IBD BUF DELTA PC 2 H
V$4187	17		IBD BUF DELTA PC 1 H
V$4193	16		IBD BUF DELTA PC 0 H
V$6132	15	4	IBD IBOPR 3 H
V$6179	14		IBD IBOPR 2 H
V$6135	13		IBD IBOPR 1 H
V$6133	12		IBD IBOPR 0 H
V$4235	11	3	IBD IBPR 3 H
V$4284	10		IBD IOPR 2 H
V$4283	09		IBD IBPR 1 H
V$4249	08		IBD IOPR 0 H
V$5161	07	2	IBD BUF FA 7 H
V$5136	06		IBD BUF FA 6 H
V$5253	05		IBD BUF FA 5 H
V$5158	04		IBD BUF FA 4 H
V$5179	03	1	IBD BUF FA 3 H
V$6120	02		IBD BUF FA 2 H
V$6119	01		IBD BUF FA 1 H
V$6121	00		IBD BUF FA 0 H

IBUF

VS symbol	Bit	Nibble	Signal name
V$5114	15	6	IBD IBUF DATA B1 7 H
V$5208	14		IBD IBUF DATA B1 6 H
V$5209	13		IBD IBUF DATA B1 5 H
V$5210	12		IBD IBUF DATA B1 4 H
V$5237	11	5	IBD IBUF DATA B1 3 H
V$5233	10		IBD IBUF DATA B1 2 H
V$5246	09		IBD IBUF DATA B1 1 H
V$5243	08		IBD IBUF DATA B1 0 H

VS symbol	Bit	Nibble	Signal name
VS$104	07	2	IED IBUF DATA B7 7 H
VS$106	06		IBD IBUF DATA B6 6 G
VS$108	05		IBD IBUF DATA B5 5 F
VS$10A	04		IBD IBUF DATA B4 4 E
VS$250	03	1	IBD IBUF DATA B3 3 D
VS$248	02		IBD IBUF DATA B2 2 C
VS$247	01		IBD IBUF DATA B1 1 B
VS$252	00		IBD IBUF DATA B0 0 A

IBXBRK

VS symbol	Bit	Nibble	Signal name
VS$1A7	04	2	-EBB IBUX BRK LIB E B
VS$1A0	03	1	EBE IBUX BRK LIE D E
VS$1B6	02		EBD IBUX BRK LIB C D
VS$1B4	01		EBC IBUX BRK LIB B C
VS$1B6	00		EBB IBUX BRK LIB A B

IDIAG

VS symbol	Bit	Nibble	Signal name
VS$1D8	05	1	IDA IBUX DIAG DONE F
VS$24d	04		ESC E DISP E D F
VS$150	03	1	ENUM XCE-E DISP E H
VS$196	02		-ICB IN MIC STALL H
VS$116	01		CA PC (SCALL H
VS1 10	00		IDPI ISTALL A G

IXLU

VS symbol	Bit	Nibble	Signal name
VS$17E	07	1	-IXU IWDX VISA 6Z E H
VS$1A6	06		IXU9 SHIFT COUNT 2 G
VS$15C	05		IXU9 VAL FIX 1 F
VS$25A	04		IXU9 SHIFT COUNT 0 E
VS$171	03	1	IXE UNLAT CUR VAL 3 D
VS$170	02		IXC UNLAT CUR VAL 2 C
VS$16E	01		IXB UNLAT CUR VAL 1 B
VS$2C6	00		IXD UNLAT LUX VAL 0 A

IOPSEL

VS symbol	Bit	Nibble	Signal name
VS8163	05	1	CSA LOWER 1 F E
VS8204	04		CSA IOPSEL 0 E H
VS8168	03	0	CSA UMISC 3 E
VS8168	02		CSA UMISC 2 E
VS8140	01		CSA UMISC 1 E
VS8130	00		CSA UMISC 0 E

IOPC

VS symbol	Bit	Nibble	Signal name
VS5129	07	2	ICA8 LD OR SAV 7 H
VS5119	06		ICA8 LD OR SAV 6 H
VS5.57	05		ICA8 LD OR SAV 5 H
VS5121	04		ICA8 LD OR SAV 4 H
VS5155	03	1	ICA0 LD OR SAV 3 H
VS5154	02		ICA8 LD OR SAV 2 H
VS515C	01		ICA0 LD OR SAV 1 H
VS5156	00		ICA8 LD OR SAV 0 H

IVABUS

VS symbol	Bit	Nibble	Signal name
VSB160	31	8	IVA BUS 31 H
VSB203	30		IVA BUS 30 H
VSB202	29		IVA BUS 29 H
VSB184	28		IVA BUS 28 H
VSB11.	27	7	IVA BUS 27 H
VSB186	26		IVA BUS 26 H
VSDJ04	25		IVA BUS 25 H
VSB115	24		IVA BUS 24 H
VSB114	23	6	IVA BUS 23 H
VSB129	22		IVA BUS 22 H
VSB11F	21		IVA BUS 21 H
VSB120	20		IVA BUS 20 H
VSB121	19	5	IVA BUS 19 H
VSB110	18		IVA BUS 18 H
VSB128	17		IVA BUS 17 H
VSB217	16		IVA BUS 16 H
VSB214	15	4	IVA BUS 15 H
VSB147	14		IVA BUS 14 H
VSB149	13		IVA BUS 13 H
VSB148	12		IVA BUS 12 H
VSB218	11	3	IVA BUS 11 H
VSB219	10		IVA BUS 10 H
VSB220	09		IVA BUS 09 H
VSB211	08		IVA BUS 08 H

VS symbol	Bit	Nibble	Signal name
VSB172	07	2	IVA BUS D7 H
VS$21C	06		IVA BUS D6 F
VSB1D5	05		IVA BUS D5 F
VS$1B6	04		IVA BUS D4 E
VSB1?8	03	1	IVA BUS D3 D
VSB1F1	02		IVA BUS D2 C
VSB161	01		IVA BUS D1 B
VSB130	00		IVA BUS D0 A

MDBUS1

V$ symbol	Bit	Nibble	Signal name
VS3A81	31	8	MD BUS D31 H
VS3232	30		MD BUS D30 G
VS?24C	29		MD BUS D29 F
VS3231	28		MD BUS D28 E
VS3252	27	7	MD BUS D27 D
VS?2F2	26		MD BUS D26 C
VS3116	25		MD BUS D25 B
VS3165	24		MD BUS D24 A
VS32B0	23	6	MD BUS D23 H
VS3234	22		MD BUS D22 G
VS?2?9	21		MD BUS D21 F
VS3235	20		MD BUS D20 E
VS3120	19	5	MD BUS D19 D
VS?2?9	18		MD BUS D18 C
VS3128	17		MD BUS D17 B
VS3283	16		MD BUS D16 A
VS3?77	15	4	MD BUS D15 H
VS3230	14		MD BUS D14 G
VS1246	13		MD BUS D13 F
VS3233	12		MD BUS D12 E
VS3121	11	3	MD BUS D11 D
VS?256	10		MD BUS D10 C
VS3119	09		MD BUS D09 B
VS3190	08		MD BUS D08 A
VS?27C	07	2	MD BUS D07 H
VS3228	06		MD BUS D06 G
VS3249	05		MD BUS D05 F
VS3290	04		MD BUS D04 E
VS1254	03	1	MD BUS D03 D
VS?257	02		MD BUS D02 C
VS3135	01		MD BUS D01 B
VS3184	00		MD BUS D00 A

MDBUSN

VC symbol	Bit	Nibble	Signal name
V$2130	31	8	MD BUS D31 H
V$2131	30		MD BUS D30 E
V$2129	29		MD BUS D29 H
V$2132	28		MD BUS D28 E
V$2140	27	7	MD BUS D27 E
V$2139	26		MD BUS D26 H
V$2138	25		MD BUS D25 H
V$2136	24		MD BUS D24 H
V$2119	23	6	MD BUS D23 H
V$2134	22		MD BUS D22 H
V$2135	21		MD BUS D21 H
V$2161	20		MD BUS D20 E
V$2164	19	5	MD BUS D19 L
V$2153	18		MD BUS D18 E
V$2162	17		MD BUS D17 L
V$2160	16		MD BUS D16 E
V$2165	15	4	MD BUS D15 E
V$2154	14		MD BUS D14 H
V$2167	13		MD BUS D13 E
V$2157	12		MD BUS D12 H
V$2158	11	3	MD BUS D11 H
V$2101	10		MD BUS D10 H
V$2166	09		MD BUS D09 H
V$2100	08		MD BUS D08 H
V$2103	07	2	MD BUS D07 H
V$2102	06		MD BUS D06 H
V$2106	05		MD BUS D05 H
V$2156	04		MD BUS D04 H
V$2107	03	1	MD BUS D03 H
V$2159	02		MD BUS D02 E
V$2200	01		MD BUS D01 H
V$2105	00		MD BUS D00 E

MEMBRD

VS symbol	Bit	Nibble	Signal name
V$C171	25	7	MCC CUOX PA ACK A H
V$C172	24		MCC OP PA ACK A L
V$C167	23	6	MCC IBF PA ACK H
V$C215	22		MBC END MAST RST DLY H
V$C173	21		MCC DIAG CODE 1 H
V$C174	20		MBC09 ARM RCO LST CYC H
V$C195	19	5	MCC PORT STAT CODE 3 H
V$C191	18		MCC PORT STAT CODE 2 H
V$C190	17		MCC PORT STAT CODE 1 H
V$C192	16		MCC PORT STAT CODE 0 H

VS symbol	Bit	Nibble	Signal name
VSA183	15	4	ICA ISTALL A H
VSA167	14		ICA IBX HSQUEST H
VSC131	13		MCC MPAC CODE OUT 1 L
VSC226	12		EBC EBOX FORK A H
VSA165	11	3	ICB OP MOP 3 H
VSA179	10		ICB OP MOP 2 H
VSA190	09		ICB OP MOP 1 H
VSA186	08		ICB OP MOP 0 H
VSA231	07	2	ICA OP ABORT A H
VSA184	06		EBC ISTALL TO MCC H
VSA153	05		EBC EBOX MOP 5 H
VSA182	04		EBC EBOX MOP 4 H
VSA174	03	1	EBC EBOX MOP 3 H
VSA185	02		EBC EBOX MOP 2 H
VSA164	01		EBC EBOX MOP 1 H
VSA154	00		EBC EBOX MOP 0 H

MBPC

VS symbol	Bit	Nibble	Signal name
VSA192	07	2	MCC1 DADR A 7 H
VSA241	06		MCC1 DADR A 6 H
VSA273	05		MCC1 DADR A 5 H
VSA274	04		MCC1 DADR A 4 H
VSA189	03	1	MCC1 DADR A 3 H
VSA170	02		MCC1 DADR A 2 H
VSA200	01		MCC1 DADR A 1 H
VSA270	00		MCC1 DADR A 0 H

MATRAM

VS symbol	Bit	Nibble	Signal name
VS3213	19	5	IBDC MAT CTX 2 H
VS3217	18		IBDC MAT CTX 1 H
VS3209	17		IBDC MAT CTX 0 H
VS3172	16		IBDC MAT TYPE 1 E
VS3163	15	4	IBDC MAT TYPE 0 E
VS3167	14		IBDC MAT MOP 1 H
VS3162	13		IBDC MAT MOP 0 H
VS3210	12		IBDC MAT CTL 1 H
VS3177	11	3	IBDC MAT CTL 0 H
VS3208	10		IBDC MAT SUSPEND H
VS3190	09		IBDC MAT BDEST MXF H
VS3211	08		IBDC MAT LAST H
VS2184	07	2	IBDT MAT DPAR H
VS3181	06		IBDR MAT EPA H
VS3180	05		IBDR MAT ADR 05 H
VS3183	04		IBDR MAT ADR 04 H

CONSOLE SOFTWARE AND COMMANDS
Default Visibility Registers

V9 symbol	Bit	Nibble	Signal name
V$0270	03	1	ISUS MAT A BUS 03 H
V$0302	02		ISUS MAT ADRS 02 H
V$0301	01		ISUS MAT ADRS 01 H
V$0300	00		IPDC GSD ALRM 00 H

OPAR

V9 symbol	Bit	Nibble	Signal name
V$4273	37	10	ICA EXECD ANUX OPAR H
V$4102	36		ICA FORCE ERUX OPAR H
V$0144	35	9	FPF OPR DAT ERR H
V$0104	34		IBUS MAT OPAR H
V$3221	33		IBCD MAT OPAR H
V$0170	32		ICAF WS OPAR H
V$XXXX	31	8	Xwrd-Filler
V$420H	30		-ICA EN OF OPAR VAL H
V$0134	2V		-IOF CP OPAR VALID H
V$8191	28		IBC OF IOC OPAR H
V$0154	27	7	ESE MBUS OPAR B3 A H
V$0155	26		ESE MBUS OPAR B2 A H
V$0114	25		ESE MBUS OPAR B1 A H
V$0155	24		EPE MBUS OPAR B0 A H
V$0184	23	6	BBB MBUS OPAR B3 C H
V$0131	22		JBC MBUS OPAR B2 C H
V$0120	21		YBB MBUS OPAR B1 C H
V$0140	20		YBE MBUS OPAR B0 C H
V$4205	19	5	ESE MBUS OPAR B3 B H
V$4123	18		ESE MBUS OPAR B2 B H
V$4213	17		EBE MBUS OPAR B1 B H
V$4200	16		EBE MBUS OPAR B0 B H
V$2126	15	4	-EBE MBUS OPAR B3 H
V$2127	14		-EBE MBUS OPAR B2 H
V$2125	13		-EBE MBUS OPAR B1 H
V$2124	12		-EBE MBUS OPAR B0 H
V$1140	11	3	MCD MD BUS OPAR B3 H
V$1149	10		MCD MD BUS OPAR B2 H
V$1147	09		MCD MD BUS OPAR B1 H
V$1148	08		MCD MD BUS OPAR B0 H
V$XXXX	07	2	Formerly MCD MD BUS OPAR B3 H
V$XXXX	06		Formerly MCD MD BUS OPAR B2 H
V$XXXX	05		Formerly MCD MD BUS OPAR B1 H
V$XXXX	04		Formerly MCD MD BUS OPAR B0 H
V$4101	03	1	IOD DBCS OPAR B3 H
V$4104	02		ISD DBUS OPAR B2 H
V$4100	01		IOD DBUS OPAR B1 H
V$4100	00		IBD DBUS OPAR B0 H

GPBOB

VS symbol	Bit	Nibble	Signal name
VSC178	31	0	OP BUS D31 F
VS0174	30		OP BUS D30 E
VSC14B	29		OP BUS D29 F
VSC166	28		OP BUS D28 E
VS0201	27	7	OP BUS D27 E
VS0260	26		OP BUS D26 F
VS0137	25		OP BUS D25 E
VSC232	24		OP BUS D24 F
VS0202	23	6	OP BUS D23 F
VS0261	22		OP BUS D22 E
VS0140	21		OP BUS D21 E
VS0220	20		OP BUS D20 E
VS0230	19	5	OP BUS D19 F
VS0172	18		OP BUS D18 E
VS0154	17		OP BUS D17 F
VS0224	16		OP BUS D16 E
VS0195	15	4	OP BUS D15 E
VS0194	14		OP BUS D14 E
VS0144	13		OP BUS D13 E
VS0289	12		OP BUS D12 E
VS0307	11	3	OP BUS D11 E
VS0300	10		OP BUS D10 E
VS0293	09		OP BUS D09 E
VS0271	08		OP BUS D08 E
VS0296	07	2	OP BUS D07 E
VS0274	06		OP BUS D06 E
VS0105	05		OP BUS D05 E
VS0275	04		OP BUS D04 E
VS0179	03	1	OP BUS D03 E
VS0264	02		OP BUS D02 E
VS0152	01		OP BUS D01 F
VS0277	00		OP BUS D00 E

DPLCOE

VS symbol	Bit	Nibble	Signal name
VS1131	07	2	ICA CPC BIT 7 A E
VS1103	06		ICA CPC BIT 6 A F
VS1238	05		ICA CPC BIT 5 A E
VS1125	04		ICA CPC BIT 4 A E
VS1048	03	1	ICA CPC BIT 3 A E
VS1232	02		ICA CPC BIT 2 A E
VS1141	01		ICA CPC BIT 1 A E
VS1241	00		ICA CPC BIT 0 A E

CPNCF

V6 symbol	Bit	Nibble	Signal name
V$A168	03	1	ICP OP MCE 3 H
V$A179	02		ICP OP MCV 2 H
V$A18C	01		ICP OP MCE 1 H
V$A186	00		ICP OP MCV 0 H

OPPORT

V6 symbol	Bit	Nibble	Signal name
V$6186	63	16	ICA OPV CTL 0 H
V$6190	62		ICA OPV CTL 1 H
V$9149	61		-ICAG OUT OPV EN H
V$5141	60	15	-ICB SRI OPV H
V$615A	59		-ICB3 IMD OPV LAT H
V$5154	58		-ICB9 PKMD IMD OPV H
V$XXXX	57		Zero-Filler
V$XXXX	56	14	Zero-Filler
V$C172	55		MCC OP PA ACK A H
V$6127	54		MCC OP PA ACK B H
V$XXXX	53		Zero-Filler
V$XXXX	52	13	Zero-Filler
V$9191	51		BBC OPBUS VALID H
V$1206	50		-PCM DRIV VAL TO PBM H
V$XXXX	49		Zero-Filler
V$C212	48	12	-ICB ALWAYS OP VALID H
V$C217	47		-ICD ALMOST OP VALID H
V$A155	46		-ICB8 ALMOST OUT OPV H
V$XXXX	45		Zero-Filler
V$A221	44	11	ICA OP ABORT A H
V$C1BJ	43		ICA OP ASURV H LAT H
V$1120	42		ICA IBF OR OP FLUSH H
V$A264	41		ICA MBOX FLUSH A H
V$C1B9	40	10	ICA OP FLUSH A H
V$4205	39		-ICA PN OP OPAR VAL H
V$0104	38		-ICP OP OPAR VALID H
V$XXXX	37		Zero-Filler
V$XXXX	36	9	Zero-Filler
V$1201	35		ICB EILL OP WRT H
V$E1C2	34		MCC RRAY OP BRC H
V$X106	33		MCC ISAP OP MUTX H
V$K135	32	8	MCC TRAP OP WRT H
V$C17E	31		-EBD9 SN OP WRT STALL H
V$8254	30		-ICAG OP REQ CIP LAT H
V$5140	29		ICB SRT OP MBY CIP H
V$6100	28	7	-ICBA OP WRT CID H
V$C212	27		-EBD9 OP WAT TM BROC H
V$C146	26		-ICD9 OP MBT INT CYC H
V$6134	25		-ICA OP WRT LAT H

VS Symbol	Bit	Nibble	Signal name
VSC170	23	5	-ICA EN OP MRT ACA H
VS5162	22		-ICA BRA OP MD RSEE H
VSXXXX	21		Zero-Filler
VS4147	20		-TCA IVA SEL OP VA Z
VS4176	19	5	-ICA BKA OP VA LD H
VS5151	18		LDC OP VA BIT 00 H
VS5241	17		ICP OP VA BIT 01 H
VSA176	16		ICB OP MCP 0 H
VSA190	15	4	TCP OP MCP 1 H
VSA179	14		ICS OP MCP 2 H
VSA168	13		TCB OP MCP 3 H
VSXXXX	12		Zero-Filler
VS4165	11	3	TCB OP MEM CTX 0 H
VSA170	10		TCB OP MEM CTX 1 H
VS4163	09		ICA OP MEM CTX 2 H
VS5143	08		TCB OP MEM REQ H
VS4145	07	2	ICB OP MEM REQ A H
VSXXXX	06		Zero-Filler
VS6162	05		-ICA SBP VA CntSEL xBU H
VS0317	04		ICI PA OP MCN RCQ H
VSXXXX	03	1	Zero-Filler
VS4172	02		TCP3 IVA SEL OP VA A H
VS4201	01		ICP5 IVA SEL OP VA B H
VS4108	00		ICP4 IVA SEL OP VA C H

PARCK

VS Symbol	Bit	Nibble	Signal name
VS0167	06	2	-MCC CMDX PA ACK A H
VS0171	05		MCC RBOX PA ACK A H
VS6169	04		XBC XB PA ACK ICM H
VSXXXX	03	1	Zero-Filler
VS5167	02		MCC IRF PA ACK H
VS0172	01		MCC OP PA ACK A H
VS6127	00		XCC OP PA ACK B H

PAKD

VS Symbol	Bit	Nibble	Signal name
VSD167	20	6	-MCC PDOX PA ACK B H
VSA186	20		CBD CntSEL TO MIC H
VSC214	21		PCC PTO MSGT BST OUT H
VSA167	20		ICA IRP REQUEST H
VSA264	19	5	ICA XBOX PINOR A H
VSA180	18		ICA IOPACT A H
VSA221	17		ICA OP ABORT A H
VSA227	16		ICA WROM PCJSR H H

VS symbol	Bit	Nibble	Signal name
V$A168	15	A	ICB DP MCF 3 E
V$A169	14		ICB DP MCF 2 F
V$A150	13		ICB DP MCF 1 H
V$PJ56	12		ICB DP MCF 0 W
V$L193	11	3	MCC PORT STAT CODE 3 H
V$C191	10		MCC PORT STAT CODE 2 H
V$C1v0	09		MCC PORT STAT CODE 1 H
V$C152	08		MCC PORT STAT CODE 0 H
V$L172	07	2	MCC DP PA ACK A H
V$51b7	06		MCC IRF PA ACK H
V$A153	05		BBC BBCX MCP 5 H
V$A1P3	04		XBC XBCX MCP 4 H
V$A174	03	1	APC BBCX MCP 3 H
V$A165	02		BBC BBCX MCP 2 H
V$A164	01		XBC XBCX MCP 1 H
V$A154	00		BBC BBCX MCP 0 H

BANK

VS symbol	Bit	Nibble	Signal name
V$A312	04	2	XAP9 BANK CONF A H
V$A193	03	1	XAP9 BANK CONF 3 H
V$a172	02		XL29 BANK CONF 4 H
V$A296	01		XAP9 BANK CONF 2 H
V$A216	00		XA29 BANK CONF 1 H

PARITY

V$ symbol	Bit	Nibble	Signal name
V$9170	63	17	BBT XCK PC FLAG H
V$U150	64		BBD SDP PB FLAG A H
V$V169	65	16	SBD ADP PA FLAG H
V$9173	62		XBD XMDP PE FLAG H
V$V174	61		SBD USTE PB FLAG H
V$9143	60		XBD MDUU PE FLAG H
V$AAAA	5F	15	Zero-Filler
V$XXXX	5E		Zero-Filler
V$XXXX	5T		Zero-Filler
V$V116	5C		EBL EDS PS LST CPC H
V$C219	5B	14	-CSB MDP PAR SNR H
V$6164	5A		CSA PAR ERR H
V$x104	59		CSAS CSA PAR H
V$P124	58		-CSB CS PAR CK A H
V$C142	57	13	CSD DSTI PAR ERR H
V$C140	56		-CSSR FLIP USTE PAR H
V$E15.	55		CSBA DATA PAR H
V$C144	54		E7P CPR PAR PER H

V$ symbol	bit	Nibble	Signal name
V$C169	47	12	-CDP RESULT PAR ERR H
V$8.31	46		DISA BYTE 10 PAR H
V$8.65	45		EDDI DISA BYTE 32 PAR H
V$8.26	44		XOPI FLIP WRNG PAR H
V$9150	43	11	XBC FLIP WRNG PAR B0 H
V$9149	42		XBC FLIP WRNG PAR B1 H
V$9145	41		XBC FLIP WRNG PAR B2 H
V$9144	40		XBC FLIP WRNG PAR B3 H
V$XXXX	39	10	Zero-Filler
V$C159	38		-RBC MCP RAM PAR ERR H
V$U140	37		XBCA MCP PAR H
V$D153	36		XBCU FLIP MCP RAM PAR H
V$XXXX	35	9	Zero-Filler
V$0210	34		PA17 GPR PPAR 00 H
V$0241	33		PA17 GPR PPAR 01 H
V$0142	32		PA17 GPR PPAR 02 H
V$0131	31	8	PA19 UNB PARITY H
V$0169	30		PSN CS PAR ERROR H
V$0105	29		DSH BREAK PAR ERROR H
V$1275	28		PM16 UNB PARITY H
V$XXXX	27	7	Zero-Filler
V$6120	26		TBD BUF DRAM PE H
V$0129	25		IBD BUF IBUF PE H
V$0201	24		ICA FORCE CPE PE H
V$0115	23	6	ICA FORCE BLOG PE H
V$0175	22		ICA ICE PE H
V$0181	21		ICE IBUF PE H
V$0103	20		ICE IBRAM PE H
V$0100	19	5	ICE BLOG PE H
V$0175	18		IDP IANUX PE H
V$0182	17		IDP IBUF PE H
V$XXXX	16		Zero-Filler
V$XXXX	15	4	Zero-Filler
V$XXXX	14		Zero-Filler
V$2165	13		MAP2 <29:4> PAR H
V$XXXX	12		Formerly MCC1 MCS PAR BER H
V$XXXX	11	3	Formerly MCC7 ACCESS PAR E
V$A118	10		MCDC D CDR PAR A H
V$A126	09		MCDC U CPE PAR B H
V$2109	08		MCCK INV CACH BIT PAR H
V$A201	07	2	MCD3 ADDR PAT PERR H
V$A251	06		-MCDU CACE DAT PERR H
V$A250	05		-MCDD WR DAT PERR H
V$A200	04		MAP2 ADDR APR PERR H
V$A2xx	03	1	MAPL TAG PERR H
V$A202	02		MAPL IAC P PERR H
V$A148	01		-MAPX IP PERR H
V$A162	00		MPC MBOX IN PE H

PSL

VS symbol	Bit	Nibble	Signal name
V$C117	20	6	EBS PSL CM IS CSD H
V$C113	19	5	-EBS PSL TP H
V$B112	18		RMD UTRAP VECTOR 4 H
V$B152	17		EBE PSL IS TO CSD H
V$A144	16		RES CURRCD 1 TO RCC H
V$A163	15	4	BBS CURRCD 0 TO RCC H
V$XXXX	14		Zero-Filler
V$XXXX	13		Zero-Filler
V$D125	12		EBE PSL IPL 4 H
V$D127	11	3	EBE PSL IPL 3 H
V$D126	10		EBE PSL IPL 2 H
V$D131	09		EBE PSL IPL 1 H
V$D130	08		EBE PSL IPL 0 H
V$XXXX	07	2	Zero-Filler
V$XXXX	06		Zero-Filler
V$C109	05		-EBS PSL IV TO CSD H
V$XXXX	04		Zero-Filler
V$9171	03	1	EDP PSL N BIT A H
V$9172	02		EDP PSL Z BIT A H
V$9166	01		EDP PSL V BIT A H
V$9167	00		EDP PSL C BIT A H

REGBUS

VS symbol	Bit	Nibble	Signal name
V$2205	07	2	REG BUS 7 H
V$2203	06		REG BUS 6 H
V$2227	05		REG BUS 5 H
V$2226	04		REG BUS 4 H
V$2224	03	1	REG BUS 3 H
V$2225	02		REG BUS 2 H
V$2204	01		REG BUS 1 H
V$2223	00		REG BUS 0 H

STALL

VS symbol	Bit	Nibble	Signal name
V$6202	28	8	ICA INSTR RDP H
V$3189	27	7	ICA IFUNK CYCLE H
V$6185	26		ICA CTX UNALIGNED H
V$5133	25		ICA6 UTRAP CTL 1 H
V$5132	24		ICA6 UTRAP CTL 0 H
V$A180	23	6	ICA ISTALL A H
V$8107	22		ICA ISTALL B H
V$C205	21		ICA ISTALL BUF A H
V$6137	20		ICA ISTALL C H

VS symbol	Bit	Nibble	Signal name
V$0197	19	5	ICA ISTALL D H
V$4156	18		ICA ISTALL G H
V$5196	17		-TOP IP FULL STALL H
V$3190	16		ICA PC ISTALL A H
V$9193	15	4	BRD BSY MUDX H
V$9180	14		ICA RTDS PE H
V$5123	13		ICA7 ICH PAR XRR H
V$9181	12		ICP IBUF PE H
V$9101	11	3	TCP IDRAM PE H
V$9176	10		IBU DAMUX PE H
V$9182	09		IDP IBMUX PE H
V$XXXX	08		Zero-filler
V$0172	07	2	MCC DP PA ACK A H
V$6127	06		MCC DP PA ACK B H
V$5167	05		MCC TBF PA ACK H
V$9162	04		MCC XBUS CS PE H
V$5221	03	1	EBD ISTALL TO ICA H
V$5106	02		EBS IBUS BUS LUN A H
V$9170	01		EBD ECS PT FLAG H
V$9110	00		EBD ECS PB LST CYC H

UPCSAV

VS symbol	Bit	Nibble	Signal name
V$B173	12	4	CSBQ UPCSAVK 12 H
V$B172	11	3	CSBQ UPCSAVK 11 H
V$B143	10		CSBQ UPCSAVE 10 H
V$B157	09		CSBQ UPCSAVE 09 H
V$B156	08		CSBQ UPCSAVE 08 H
V$B15C	07	2	CSBQ UPCSAVE 07 H
V$B171	06		CSBQ UPCSAVE 06 H
V$B174	05		CSBQ UPCSAVE 05 H
V$B17C	04		CSBQ UPCSAVE 04 H
V$E149	03	1	CSBQ UPCSAVE 03 H
V$E141	02		CSBP UPCSAVE 02 H
V$E142	01		CSBP UPCSAVE 01 H
V$E148	00		CSBP UPCSAVE 00 H

MBUS

VS symbol	Bit	Nibble	Signal name
VS4231	30		XBUS D30 E
VS4227	29		XBUS D29 E
VS4223	28		XBUS D28 E
VS4107	27	7	MBUS D27 E
VS4237	26		XBUS D26 E
VS4211	25		XBUS D25 E
VS4236	24		XBUS D24 E
VS4231	23	6	XBUS D23 E
VS4232	22		XBUS D22 E
VS4206	21		XBUS D21 E
VS4204	20		XBUS D20 E
VS4167	19	5	XBUS D19 E
VS4131	18		XBUS D18 E
VS4127	17		XBUS D17 E
VS4120	16		XBUS D16 E
VS4247	15	4	XBUS D15 E
VS4243	14		XBUS D14 E
VS4244	13		XBUS D13 E
VS4245	12		XBUS D12 E
VS4122	11	3	XBUS D11 E
VS4274	10		XBUS D10 E
VS4276	09		XBUS D09 E
VS42C2	08		XBUS D08 E
VS4186	07	2	XBUS D07 E
VS4182	06		XBUS D06 E
VS4138	05		XBUS D05 E
VS4177	04		XBUS D04 E
VS4196	03	1	XBUS D03 E
VS4101	02		XBUS D02 E
VS4100	01		XBUS D01 E
VS4188	00		XBUS D00 E

CHAPTER 11

WBOX

11.1 WBUS

The WBus, or Write Box, is used by the EBox, IBox, and FBox to write GPR/Scratch Pads, or to transfer result or operand data. Figure 11-2 shows how the GPR/SP registers receive WBus data and parity for each box. This figure also shows the print out location of where data is driven onto the WBus.

Figure 11-3 shows the WBus termination, and provides a closer look at the WBus drivers and receivers.

Table 11-1 lists the WBus pin location and WIRES for each of the WBus data bits. Table 11-2 shows the WBus pin location and WIRES for each of the WBus byte parity bits.

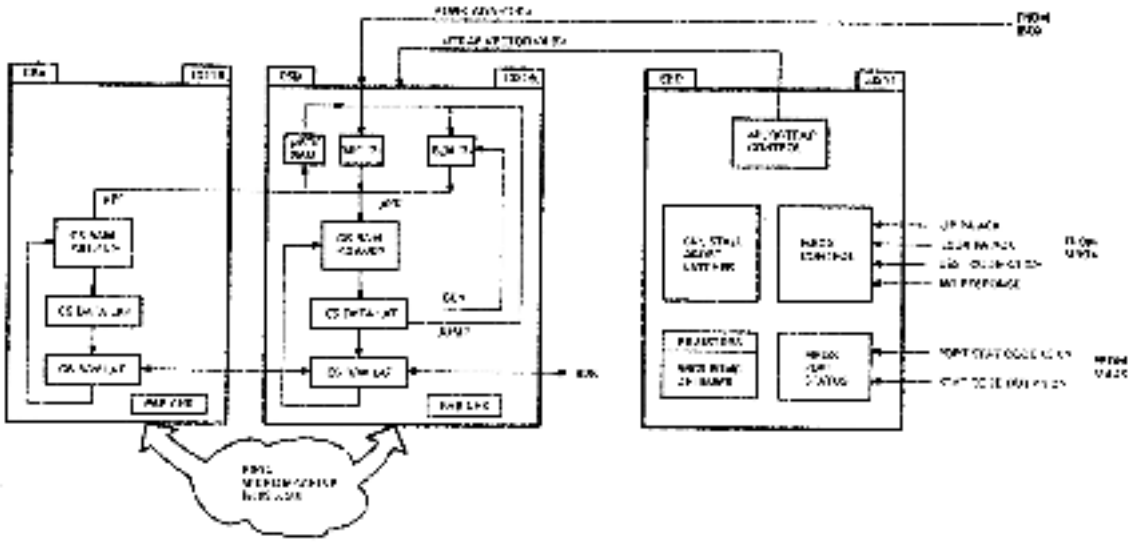


Figure 11-1 8800x Nonvolatile Block Diagram (Sheet 1 of 2)

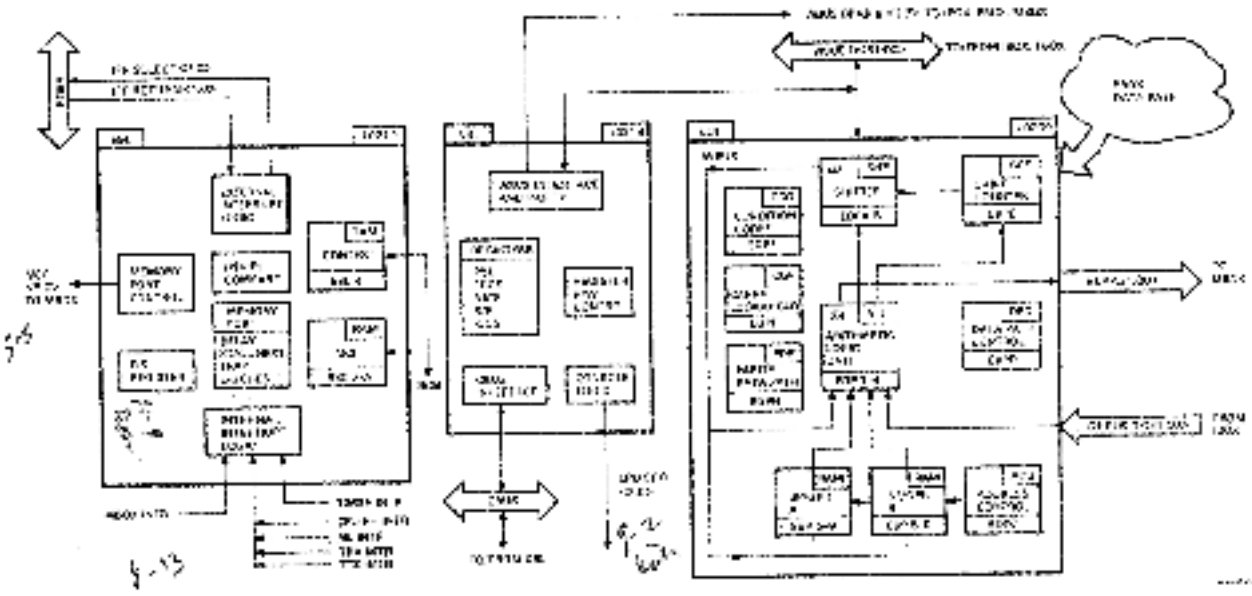


Figure 1-1 8086 Modules Block Diagram (Sheet 2 of 2)

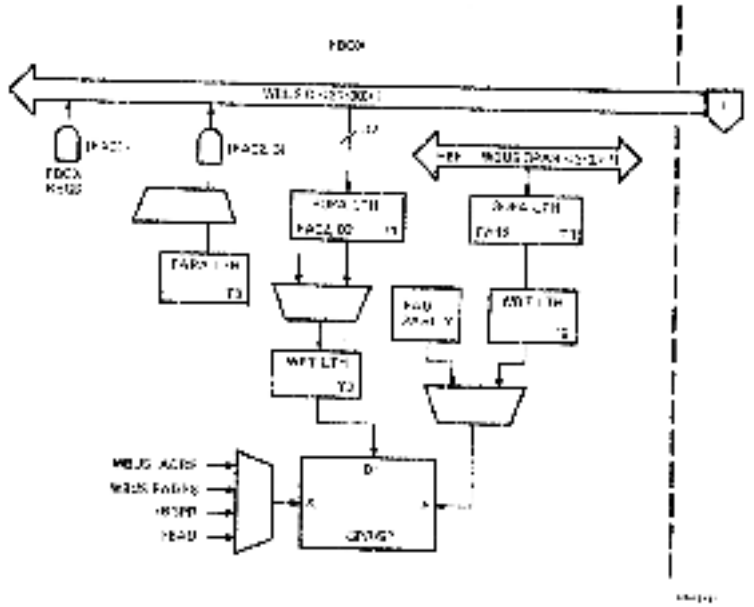


Figure 11-2 WBus Data Path (Sheet 1 of 2)

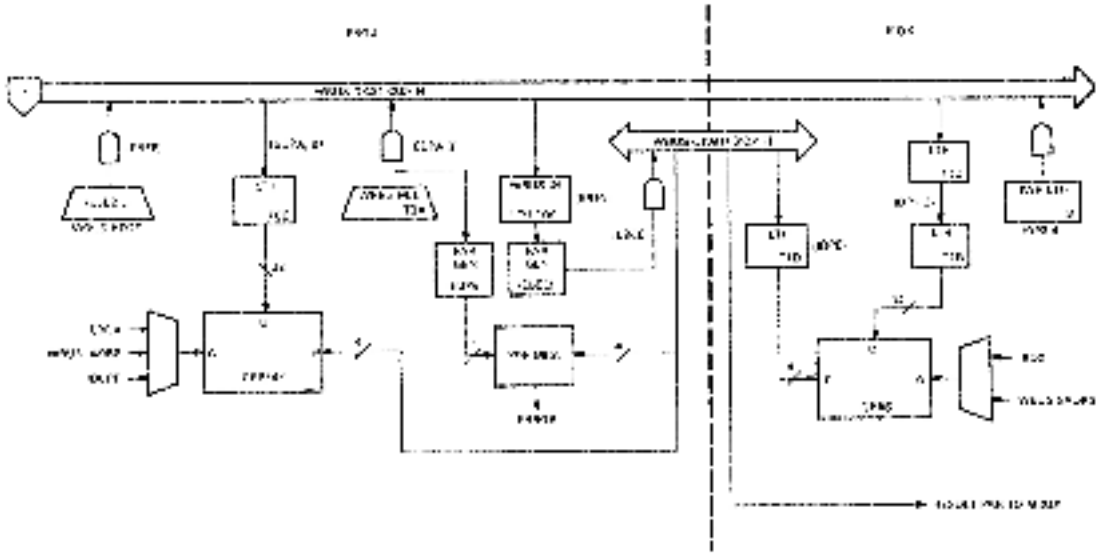


Figure 11-2 MBus Data Path (Sheet 2 of 2)

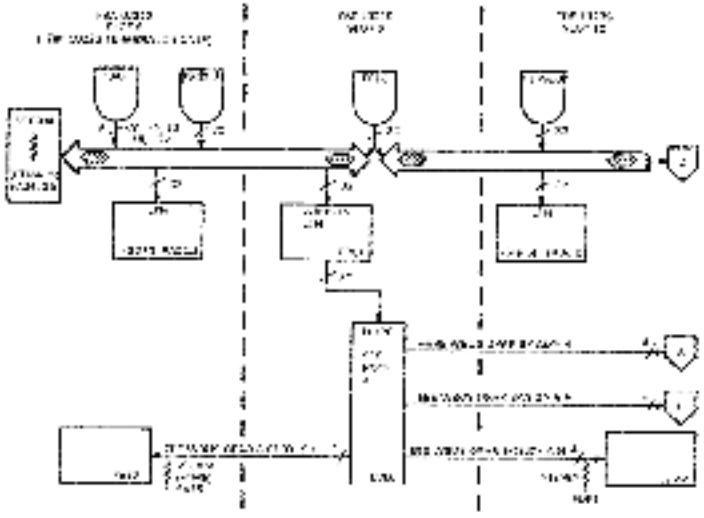


Figure 11-3 Wbus Physical Distribution (Sheet 1 of 2)

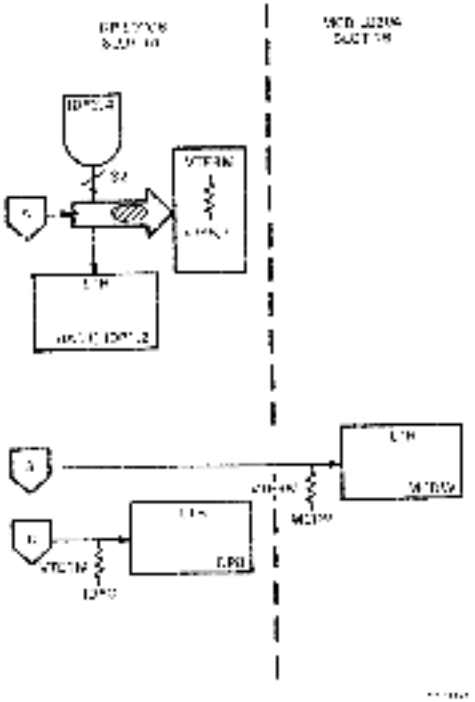


Figure 11-3 WBus Physical Distribution (Sheet 2 of 21)

11.1.1 WBus Signal Pin Location

Table 11-1 WBus Signal Pin Location

Signal Name	WDUS D<31:00> A B			WBUS D<31:00> B E			
	ESE 5	PSA 8	V*KRM	SBB 9	EDP 10	IDP 14	VTSEN
WBUS D00	A09	A10	V$0036	A10	A46	A35	V$4186
WBUS D01	A03	A04	V$0161	A01	A03	A21	V$4170
WBUS D02	C05	C06	V$0263	C06	C27	A25	V$4131
WBUS D03	C07	C08	V$0267	C08	C60	A31	V$1194
WBUS D04	A11	A12	V$0294	A12	A27	A05	V$4177
WBUS D05	A05	A06	V$0140	A06	A07	A5.	V$4138
WBUS D06	C07	C08	V$0197	C08	C1.	A07	V$4182
WBUS D07	C09	C00	V$027.	C00	C02	A7	V$4184
WBUS D08	A17	A18	V$0208	A18	A25	A47	V$4202
WBUS D09	A15	A16	V$0273	A15	A06	A07	V$4276
WBUS D10	C11	C12	V$0341	C12	C09	A67	V$4174
WBUS D11	C03	C04	V$0268	C04	C51	C75	V$4120
WBUS D12	A19	A20	V$0231	A20	A49	409	V$4240
WBUS D13	A7	A22	V$0216	A10	A27	015	V$4244
WBUS D14	C25	C20	V$0267	C40	C11	021	V$4243
WBUS D15	C09	C70	V$0270	C70	C73	631	V$4247
WBUS D16	A45	A46	V$0230	A46	A47	B25	V$4128
WBUS D17	B45	A29	V$0.17	B32	A85	B52	V$4127
WBUS D18	B53	C17	V$0x62	B04	C29	869	V$4131
WBUS D19	C73	C74	V$0177	C74	C71	873	V$4157
WBUS D20	A51	A54	V$0104	A54	B29	C09	V$4253
WBUS D21	A33	A24	V$0162	A04	A60	C19	V$4166
WBUS D22	C41	C42	V$0196	C42	C15	C25	V$4252
WBUS D23	C77	C78	V$0116	C76	C57	C35	V$4251
WBUS D24	A35	A36	V$0107	A56	A41	C07	V$4256
WBUS D25	A65	A66	V$0100	A66	A61	C17	V$1271
WBUS D26	C45	C46	V$0198	C46	C25	027	V$4207
WBUS D27	C05	C06	V$0260	C46	C67	C37	V$4107
WBUS D28	A59	A60	V$0105	A50	A71	C45	V$4223
WBUS D29	B07	B58	V$0'66	A60	A71	C53	V$4207
WBUS D30	C5'	C52	V$0175	C52	C17	C65	V$4221
WBUS D31	C87	C60	V$0201	C88	C50	C72	V$4232

Table 11-2 Distribution of WBus Byte Parity Signals

WBUS Signal Name	ESE 09	MCU 16	KDP 10	IDP 14	PBA 08	VTSEN
~KDK WBUS OPAR B0 H	A52	012				V$2 15
EBE WBUS OPAR B0 B C	A57		C78			V$8157
E H WBUS OPAR B0 B E	A58			A41		V$4204
EBE WBUS OPAR B0 C E	A51				A75	V$0120
~EBE WBUS OPAR B1 H	A62	B25				V$2125
EBE WBUS OPAR B1 A B	A73		C30			V$8134
EBE WBUS OPAR B1 B H	A67			A40		V$4210
EBE WBUS OPAR B1 C E	A67				A05	V$0129

Table 11-2 Distribution of MBus Byte Parity Signals (Cont.)

MBUS Signal Name	EBB 09	MCD 16	EDP 10	IDP 14	FBA 08	Wterm
-EBE MBUS OPAR B2 E	B07	B27				V62127
-EBF MBUS OPAR B2 A 0	B10		C35			V60156
EBD MBUS OPAR B2 D 3	B10			A49		V64123
-EBE MBUS OPAR B2 C 0	B12				A03	V60131
-EBE MBUS OPAR B3 E	B04	B30				V60126
EBB MBUS OPAR B3 A 4	B08		C30			V68156
-EBF MBUS OPAR B3 D 1	B11			A39		V64205
EBE MBUS OPAR B3 C 4	B11				A53	V60186

NOTE:

The defined visibility register "MBUS", B01A, in the "B-side" of the MBUS (that is, terminated on the IDP module).

11.2 EBOX GPR/SCRATCH PAD USAGE

Table 11-3 shows the usage of EBox GPR Scratch Pad registers in the KA86. If a more detailed listing is needed, refer to DEF.MIC in the EBox microcode listing, KA8600.MCD or KA8650.MCD.

There are a total of 256 32-bit registers, with two copies of each for parity redundancy. If there is a parity error in the "A" set of the GPRs, the "B" set can be copied into the "A" set.

Table 11-3 EBox GPR/Scratch Pad Usage

Usage/Assignment	S	Scratch Pad Addresses) (EXAM/ESC)
General Purpose Registers	R0 - R11	0 - 11
	AP	12
	FP	13
	SP	14
Temporaries	T0 - T1b	10 - 2F
Secondary Temps or Misc		30 - 3F
Constants		40 - 4F
Block of Zeros		50 - 5F
CSM Registers		C0 - C8
Extra Temps		CA - CD
KA86 Status Registers		CF - DF
VAX Architectural Registers		E0 - EE
USR Keywords Names		EL - YE
Microcode Stack		F0 - FF

11.3 EBOX MICROWORD

A worksheet is provided for the EBox microword, followed by a description of the EBox microword. The physical bit numbers are determined by the RAM slots on the CSA and CSB boards. The logical bit position is the bit positions as they would be displayed for an E/BCS "Adzs" Console command.

11.3.1 EBox Control Store

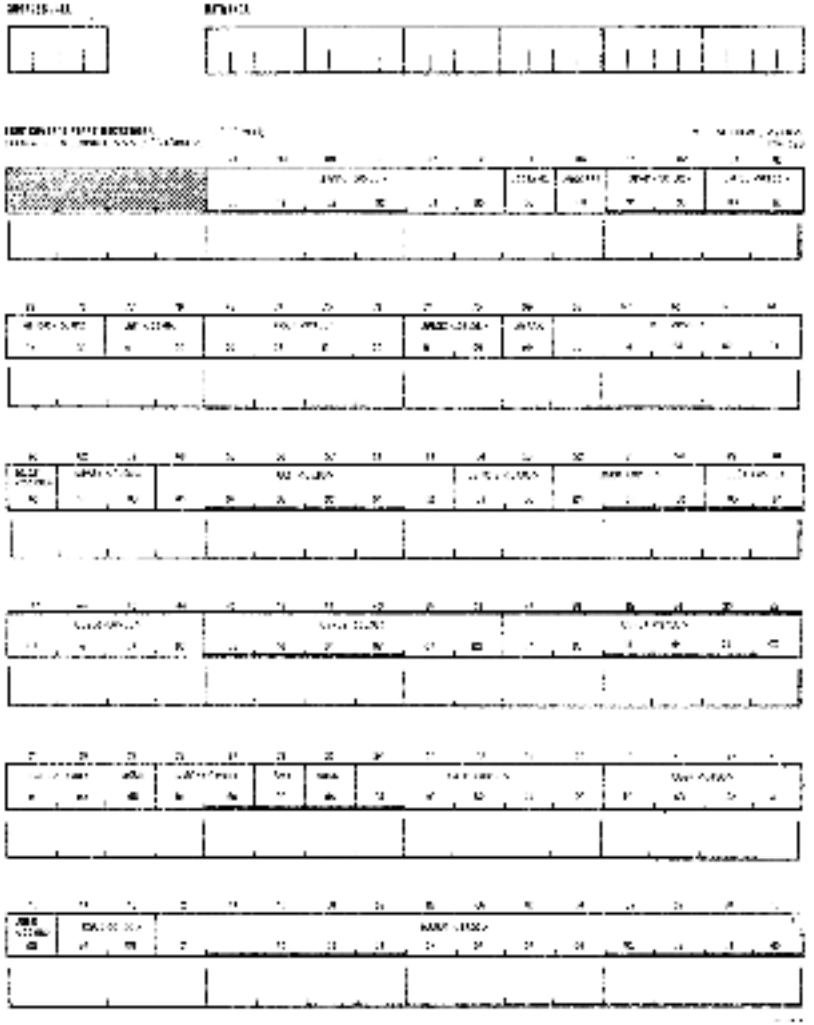


Figure 11-4 EBox Control Store Microword Worksheet

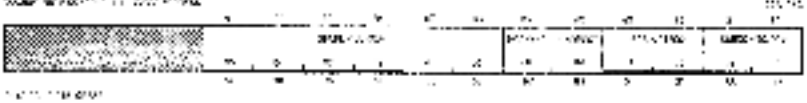


Figure 11-3 FBox Microword <91:86>

SPARE FIELD

The SPARE field contains the three bits of the microword. There are 62 bits in each SBox microword. Currently, only 56 bits (microword <85:00>) are used to control the FBox and CPU.

OCCSYNC<00>

The MICRO CONDITION CODE SYNC field indicates that the PSL condition codes are valid. The FBox uses this field to know when it can use the condition codes for branching.

LNOCHST

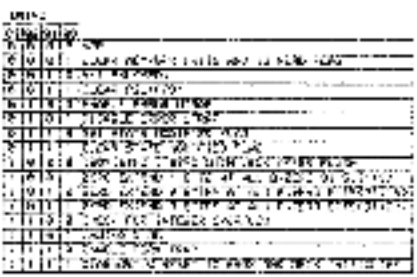
This field notifies the hardware that the SBox will request Low Mode.

LPAR<01:00>

The MICRO PARITY field contains the parity for the microword. LPAR<1> and LPAR<0> cover mutually exclusive sets in the SBox microword.

LMISC<03:00>

The MICRO MISCELLANEOUS field provides those miscellaneous functions which do not require the assignment of an entire field for control.



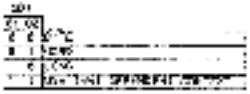
EBOX
Ebox Control Store



Figure 11-4 EBox Microword <79:64>
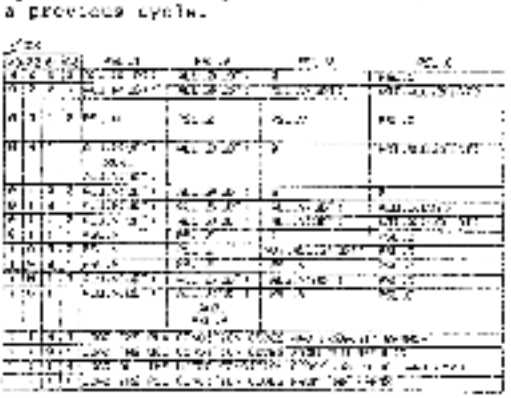
DST<01:00>

The MICRO DATA TYPE field is the EBox data type specifier. The data type is used for EBox initiated memory reference and condition code functions. If data type OCTA or OCTD is desired, the instruction dependent data type must be used.



DCOR <03:00>

The MICRO CONDITION CODE field controls loading of the ALU and microcondition codes. In the following condition code table, the values for ALU.x are a function of the ALU result in the current cycle and are independent of the ALU condition codes latched from a previous cycle.



JEBOX <01:00>

The MICRO EBOX field is used to control and synchronize EBox operations with the EBox.



UMARK<00>

The MICRO MARK field is used for setting breakpoints in the microcode to assist in micro program debug.

UNCF<05:00>

EM2 MICRO MEMORY CONTROL FUNCTION field specifies a memory function <03:00>, or a register select function <02:00>.



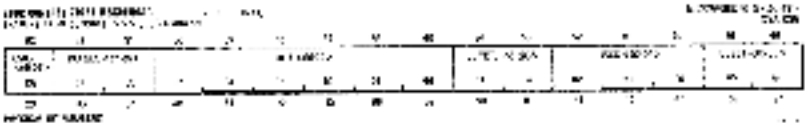
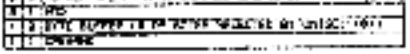


Figure 11-7 EBox Microword <h:40>

MDPSEL<01:00>

The MICRO DESHARD SELECT field is used to control the source of data the IBox will transfer over the DP Bus.

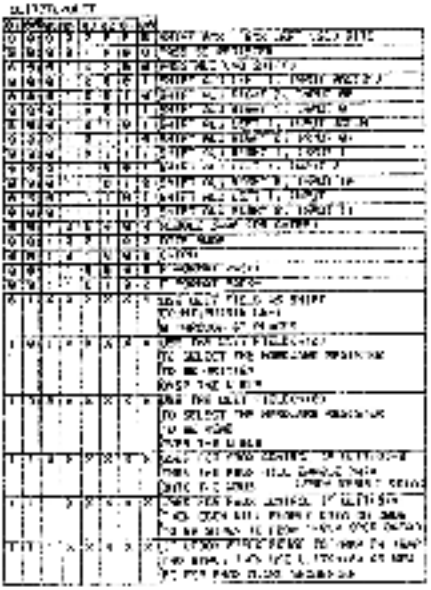


UCTL<05:00>

The MICRO LITERAL field provides the value for each configuration of the ULITCTL field.

ULITCTL<01:00>

The MICRO LITERAL CONTROL field controls the use of the MICRO LITERAL field.



UB<CR<02:00>

The MICRO SHIFT COUNTER CONTROL field controls functions involving the SC and STACK registers. Incrementing or decrementing the SC, and loading the SC from the UNIT field affect shifting operations in the current cycle. Branch tests always use the value of the SC or STACK registers at the beginning of the cycle.



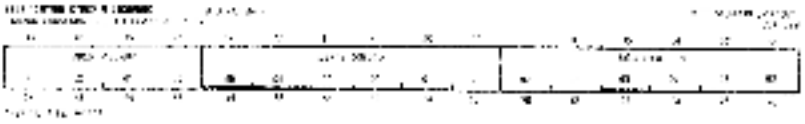


Figure 11-8 SBox Firmware (4/5)

LDBSY<09:00>

USNC2<07:00>

USNC1<07:00>

The MICRO SOURCE 1, MICRO SOURCE 2, and MICRO DESTINATION fields are used to control source/input buses and destination addressing. Also encoded in these fields are functions to manipulate SYCON and the SBox stack pointer (STP).

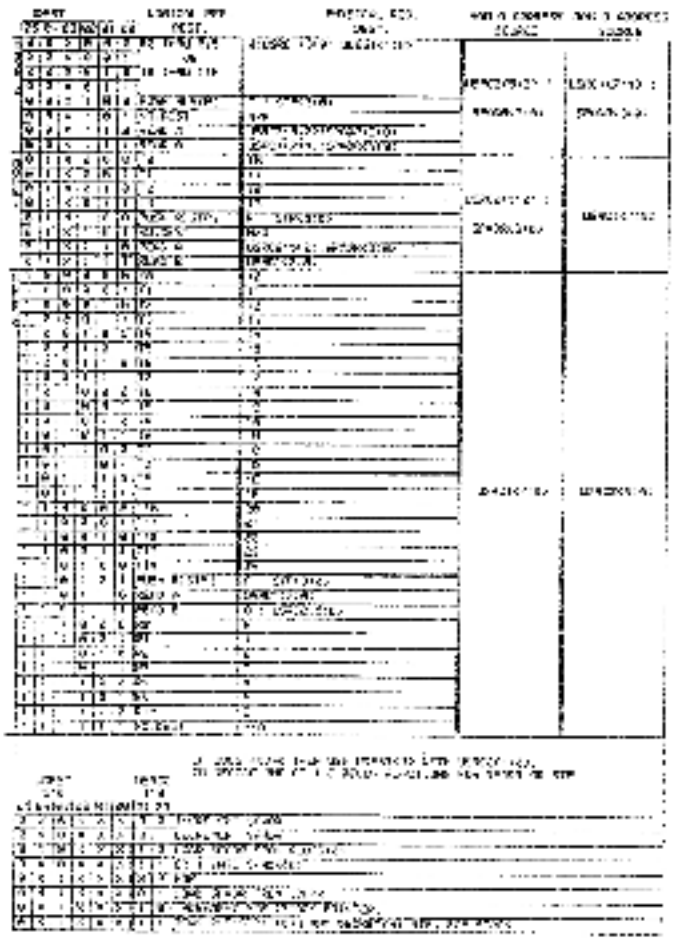




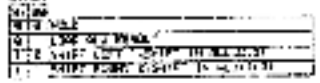
Figure 11-9 KA65x Microword <51:16>

SMI<00>

The START MACRO INSTRUCTION field is used to indicate the start of a new macroinstruction. It is asserted low in the first cycle of a macroinstruction, and deasserted at all other times.

VVVQX<01:00>

The MICRO VMS REGISTER CONTROL field controls the loading and shifting of the VMS register.



QUMX<00>

The MICRO B MULTIPLEXER field selects the scratchpad location addressed by RB if "0" or the DP bus (if "1") as the B input to the ALU.

QAMX<00>

The MICRO A MULTIPLEXER field selects the scratchpad location addressed by RA if "0" or VM2 (if "1") as the A input to the ALU.

QAIU<04:00>

The MICRO ARITHMETIC LOGIC UNIT field specifies any one of 32 arithmetic or logic operations for the ALU.



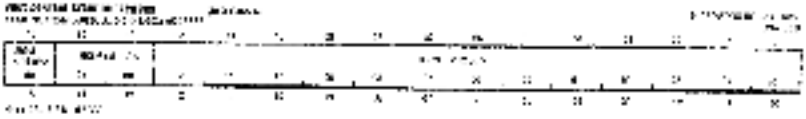
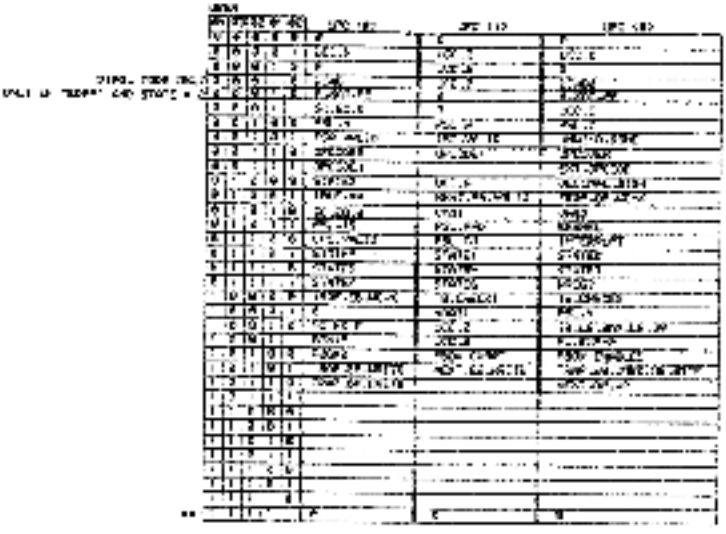


Figure 11-10 XBox Microword <15:00>

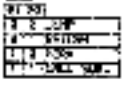
CBEN<04:00>

The MICRO BRANCH ENABLE field specifies one of 23 possible branch enable functions. Each branch enable can provide an 8-way branch. The results are ORed into the three low-order bits of the next uPC.



CSUBC<1:00>

The MICROSUBSEQUENCER CONTROL field specifies how the uPC will be formed.



UJUMP <12:00>

The MICRO JUMP field generally specifies the next microaddress. Depending on the USUB field content, there are other components that get ORed with the UJUMP field to produce the actual microaddress.

11.3.2 EBox Context

The Context RAMs are loaded by the console over the SDI, and contain native mode instruction dependent contexts. Figure 11-11 shows the format of the Context RAM microword. Table 11-4 lists the bit descriptions for the context RAM.

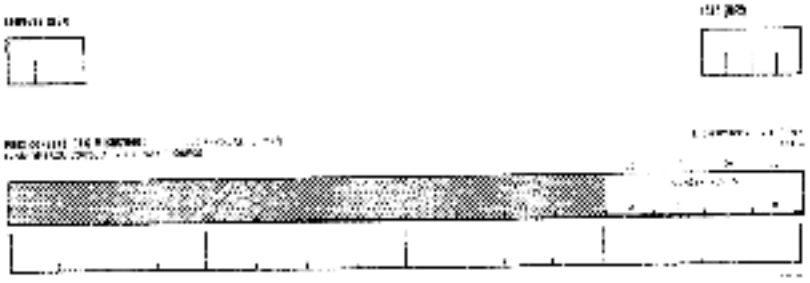


Figure 11-11 EBox Context (CTA) Microword Worksheet

Table 11-4 Context <3:0>

0 = Byte
2 = Longword
3 = Quadword
4 = Octaword
9 = Word

11.3.3 Memory Control Function (MCF)

The MCF RAMs provide a lookup table to decode the EBox MCF opcode field. The PROMs are addressed by the MCF field, and are loaded and verified by the console over the SDI. The decodes provide various control functions throughout the EBox. Figure 11-12 shows the layout of the MCF microword, and Table 11-5 contains the bit descriptions for the MCF RAM microword.



Figure 11-12 MCF Microword Worksheet

Table 11-5 MCF Microword Bit Description

1. <15> WRITE Check - This bit is asserted if the uMCF field spec= f es a memory command for which write access checking is performed.

2. <14> -XCR REQ - The default for this bit is no XCR Req. It will be asserted (low) for those encodings of the uMCF field for which EBox DA ACK is expected from the XBox.

3. <13> IN XBox - This bit will be set for any uMCF that provokes XBox activity.

4. <12> XCR PAR - Odd parity over XCR RAX field.

5. <11> -LOAD XBXC REG - Asserted (low) for those uMCF commands which must be loaded into the Next Register.

6. <10> -LOAD TRAP REG - This bit is asserted (low) for those uMCF commands which must be loaded into the Trap Register.

7. <09> -REQUEUE NEXT REF - This bit is asserted (low) if the EBox uMCF field specifies the Requeue XBox Port Next Reference function.

8. <08> -REQUEUE TRAP REF - Another low asserted bit, -Req= if IRAD REF will be asserted if the uMCF field specifies the Requeue EBox Port Trapped Reference function.

9. <07> -CLR IB DS - This bit will be asserted (low) during the Requeue Instruction Buffer Port Reference or Requeue Operand Port Reference (on behalf of IBuff) to reset the memory port status code for the IBuffer port.

10. <06> -CLR OP PS - This bit will be asserted (low) if the uMCF field specifies the Requeue Operand Port Reference or Requeue Operand Port Reference with No Cross Page Boundary Checking to reset the OP PORT port status.

11. <05> -CLR EB DS - This bit is asserted (low) if the uMCF field specifies the Clear EBox Port Status function to reset the EBox Port status latches.

12. <04> E DISP 1 - This signal is asserted if the uMCF field specifies the XBox Dispatch IBox function. This function is used by diagnostic microcode to test the IBox. The XBox will dispatch its Microcode based on the contents of the EBox uMCF in the Microinstruction following the Microinstruction that has a uMCF equal to E DISP 1.

13. <03> -NEW REQ - This bit is asserted (low) for those encodings of the uMCF for which both EBox DA ACK and EP YD RDY= are expected from the XBox.

14. <02> NEW WRT - If -NEW REQ is asserted, this bit will be set for those uMCF encodings that are OP FORD memory write commands.

15. <01> -OP WRT - This bit is asserted (low) if the uMCF field specifies the Operand Port Write Function.

16. <00> IN OP WRT STL - This signal is asserted if the uMCF field specifies the Operand Port Write Function or the Invole Operand Port Write Stall Function.

11.4 EBOX MICROTRAP VECTOR ASSIGNMENTS

The EBox Microtrap Vector assignments are listed in Table 11-6.

Table 11-6 EBox Microtrap Vectors

Vector	Microtrap Condition	Action
02	PBox Problem	The PBox service routine determines if an PBox hardware error occurred and if so, sets the request for PBox service in SHM.STS, then it passes control to SHM.
04	MBox Error Register	The MBox interrupts the EBox when it latches the MBox error address register and wants it saved. SHM reads the register and places it in scratchpad MBAR.SAV, calls interrupt code, then rolls back pending instructions and restarts at PC interrupted.
06	Interrupt	Interrupt requests are filtered by MBox hardware and the request with highest priority is latched in CSIINT and a microtrap to interrupt microcode is generated. If the IR1 is 10 then the MBox has latched an error. Interrupt microcode notices this fact and sets EBM.STS <07>, the MBox service request and branches to SHM.
08	EBox Hardware Error	When the EBox detects a hardware error, it latches the reason in EUR2 and EBCS and generates a microtrap at vector 8.
0A	XBox TB Error	When the MBox detects a problem with the TB during EBox port requests it reports the event with a Port Status Code = 8 and destination code = 2. The error is serviced by SHM.
0C	MBox Fatal Error	Errors which cause unpredictable results in the MBox for any type of request are reported to the EBox the same as EBox errors are reported.
10	IBox Hardware Error	An IBox hardware error is latched in the TAR and the IBox stalls. Other IBox errors form an IBox microtrap request; if S DMA is set, EBox hardware services the request when it finishes current execution and attempts to get operands from the IBox (fork).

11 - 17	MBox TB Error	If the MBox returns an OP PORT status code of other than no problem for a memory request issued by the OP PORT, the microtrap vector will be in the range of 11 - 17, with the least significant bits being supplied by the least significant bits of the OP PORT status latches.
18	IBox Hardware Error	When IBF PE, DRAM PE, or R-MUX PE is detected, the IBox stalls the IBUF PORT, and sends the error flags to the EBox where they are latched in IRS. If a flush command is executed before the EBox forms for an operand, the error will be cleared. Otherwise, it will be reported through this vector.
19 - 1F	MBox TB Error	If the MBox returns an IBUF PORT status code of other than no problem for a memory request issued by the IBUF PORT, the microtrap vector will be in the range of 19 - 1F. The least significant bits of the vector will be provided by the least significant bits of the IBUF PORT status latches.
1x	MBox Sync	After the EBox flushes and starts the IBUFF, it stalls until the CPC is valid. CPC VALID indicates that a new macro instruction is being processed. If an IBox error occurs which inhibits loading of CPC, a microtrap will be generated through this vector.
1F	Unwind Blog	If an IBox error occurs during the Blog Control command it will be reported through this vector.

11.5 EBM FATAL ERROR LOOPS

When EBM detects a condition which it can't handle or report to another part of microcode, it branches to a subset loop and waits for the console to detect a Keep Alive Ceased condition. Console software checks for the micro PC's in Table 11-7 and if found, reports a corresponding message to describe the FAE.

Table 11-7 EBM Fatal Error Loops

EBox uPC	Reason
20	RAM A and RAM B parity error in the same cycle
21	Multiple EBox errors (two errors detected on EBM entry vector 6)

11.6 CONSOLE SUPPORT MICROCODE (CSM)

CSM utilizes a 32 location section of SPox microcode (addresses 1000 - 109F) for the execution of console commands. When required, the console will load the appropriate routine into the one KDS/CSM overlay region, then pass control to CSM to execute the routine.

CSM.MIC contains the resident CSM source code and overlay (non-resident) areas. CSM.MIC can be found in KA68U0.MCR or KA68U0.MCR, and contains:

1. DC021 read and write subroutines
2. Packet read and write subroutines
3. Command and response read and write subroutines
4. Certain resident entry points
5. Command fetch, checksum calculation microcode
6. Console wait loop

Table 11-8 lists the special CSM microaddresses, and Table 11-9 lists the CSM overlays.

Table 11-8 Special CSM microaddresses

Address	Contents
1000	The start of the non-resident section
1001	The restart address of the Find_64 Kb and Find_RPB procedures
1007	The jump-data used to tell the console that certain special commands have finished (e.g., the end of power-on part #1)
0030	The start of the Compatibility Mode microcode
1005	The microaddress to send a 1-packet response
1000	The microaddress to send a 2-packet response
1001	The microaddress to start housekeeping after having loaded CSM.STATUS
137C	The entry into XDT microcode, used by the TRD command
0402	The start address of CSM.CMSG_READ
0041	The start address of CSM.CMSG_WRITE
0400	The start address of SSM.MREG.XX, needed for some CSM commands
0040	The start address of PR.SNDATTRMTE.TM
0042	The start address of MTPP.CACHK.SNEED
0044	The start address of READ.PROX.REC
0047	The start address of PROX.WRSEL.D
0010	The start address of REQUEST.SOFTWARE.INTRPT
0265	The start address of CRT.PCP.SYRRDUTINE
0279	The start address of LOL.DTE.2
09C2	The start address of MTPR.TCCS.SUB
09C4	The start address of MTPR.CSL.XPRTX
09C8	The start address of MTPR.CSL.WRITE.1

Table 11-9 CSM Overlays

CSM001	Read IPR Group 0 (KSP,ESP,SSP,USP,ISP,P0BR,P0LR,P0BR,P1BP,P1LR, SCBB,SCR,PIE)
CSM002	IRD
CSM003	Continue Microflow
CSM004	Examine Ebox Scratchpad Register
CSM005	Examine Ebox Misc. Register (SPADR, STATE, IRDPR)
CSM006	Examine Ibox Misc. Register (EXD, VA.SAV, VIBA.SAV, CPC, ISS, ESA, VPCTETS)
CSM007	Deposit Ebox Scratchpad Register
CSM008	Deposit Ebox Misc. Register (SPADR,STATE)
CSM009	Examine Physical CPU Memory
CSM010	Flush Ibox and Icap
CSM011	Deposit Physical CPU Memory
CSM012	Translate Test Virtual Read
CSM013	Find 64kb
CSM014	Read Internal Mbox Register
CSM015	Read Internal Ibox Register (CSLINT, PSL, SDESP, IDC, ESCS, SDES)
CSM016	Write Internal Mbox Register
CSM017	Write Internal Ibox Register (CSLINT, PSL, SDMC, EBCS)
CSM018	Clear Internal Mbox Register
CSM019	Translate Test Virtual Write
CSM020	Examine PAMM Location
CSM021	Flush IBOX (Set do not loop)
CSM022	Access ICA
CSM023	Find KPM
CSM024	Examine PC
CSM025	Read IPR Group 1 (IPL, ASTLVL, IDPR, SIRR, SISR, ICR, KDBR, ICCS, PAMLOC, SID, CSRD)
CSM026	Read IPR Group 2 (RXCS, RXDB, PMR, TXCS, TXDB)
CSM027	Read IPR Group 3 (SCXCS, PAMACC, ACCS, ESPD, STXDB)
CSM028	Write IPR Group 0 (KSP, ESP, SSP, USP, ISP, P0BR, P0LR, P0BR, P1BR, P1LR, SCBB, SBR, SLR)
CSM029	Write IPR Group 1 (IPL, ASTLVL, SIRR, SISR, MTCR, RCCS)
CSM030	Write IPR Group 2 (RXCS, RXI, PMR, PAMLOC, TXCS, PAMACC, TXDB, PMR, PCCS)
CSM031	Write IPR Group 3 (MAPEN, VMLA, TBIS, IBLPR, CSWP, CRBT)
CSM032	Write IPR Group 4 (SCXCS, SCXDB, ESCA, ESPD)
CSM033	Read IPR Group 4 (NDECC, NEMA, NDCTL, NCCTL, NEBG)
CSM034	Write IPR Group 5 (MDECC, MEMA, NDCTL, NCCTL, NEBG)
CSM035	Write IPR Group 6 (IDPR, TXCS)
CSM036	Clear Main Memory
CSM037	RESERVED FOR FUTURE GROWTH
CSM038	RESERVED FOR FUTURE GROWTH
CSM039	RESERVED FOR FUTURE GROWTH
CSM040	CSM.SW?XX.PO PART 1
CSM041	CSM.SW?XX.PO PART 2
CSM042	CSM.SW?XX.MICRO
CSM043	(formerly CSM?21) Flush_Ibox and IRD without NOPc

NOTE

To determine which CSM overlay was last loaded, use the following procedure:

1. Determine the TLL memory address where the overlay number is stored using the SHOW VERSION console command.

2. Examine the TLL address using the EXAM console command.

3. Convert the octal value to decimal to determine the CSM overlay loaded.

Example 11-1 Determining which CSM overlay was loaded

```
1.  >>>>SHOW VERSION
        CSMOVN:013251

2.  >>>>E/L/S   #D13251          ;USE   %O   TO   SP<C|C>   OCTAL
        C 013251 000017

3.  17 (octal) = 15 decimal, therefore the last CSM overlay
    loaded was CSM015.BPN
```

11.7 EBOX MICROCODE LOCATIONS/REGIONS

Table 11-10 contains a list of EBox microcode locations/regions.

Table 11-10 EBox Microcode Locations/Regions

Address	Contents
0001	Integer overflow trap after TRD
0007	Trace trap entry point
0006	EBox TB miss routine
000B	Memory management faults (Access violation or translation not valid)
000C	M-Bit not set for EBox references
000E	XBus page boundary crossing
000F	Unaligned reference traps
0012	DP PORT TB miss routine
0014	M-Bit not set for DP PORT references
0015	Entry DP PORT write and DP PORT I/O read (combined)
0016	Page boundary crossing in DP PORT
001A	IBUF TB miss routine
001D	IBUF read from I/O space
0040	Invalidate TB routine
0060	Routine to fetch PTE from Page Tables
0800-107F	Native mode C and E Forks
1A00-1A7F	Native mode D Fork
1B00-1B7F	Native mode A Fork
1C00-1FFF	User microcode (UCS)

11.8 EBOX UPC TESTPOINTS

Table 11-11, a list of the EBox UPC backplane pins, may be used for scoping or used with a logic analyzer. For triggering, use *CLOCKS 141 D E* and enable with *LD EKA L*.

Table 11-11 KBQX Micro PC Testpoints

Channel	Signal Name	Print	Section/Slot/Pin
0	KUPC 00 H	C8P	A03.6E
1	EUPC 01 H	C8E	C03.06
2	KUPC 02 H	C8D	CC3.42
3	EUPC 03 H	C8C	BC3.03
4	KUPC 04 H	C8B	BC3.48
5	EUPC 05 H	C8D	B03.10
6	EUPC 06 H	C8B	B/1.56
7	EUPC 07 H	C8A	B03.06
8	KUPC 08 H	C8B	B01.5\
9	EUPC 09 E	C8H	H03.08
A	KUPC 10 H	C8D	H03.54
B	EUPC 11 C	C8B	B01.06
C	PUPC 12 E	C8D	B03.48
CQ	LD EKA L	C8B	A01.44
CLK	CLOCKS 141D E	C8H	B03.44
MARK	EBOX MARK H	C8A	CU3.69 or H11.45

CHAPTER 12

FBOX

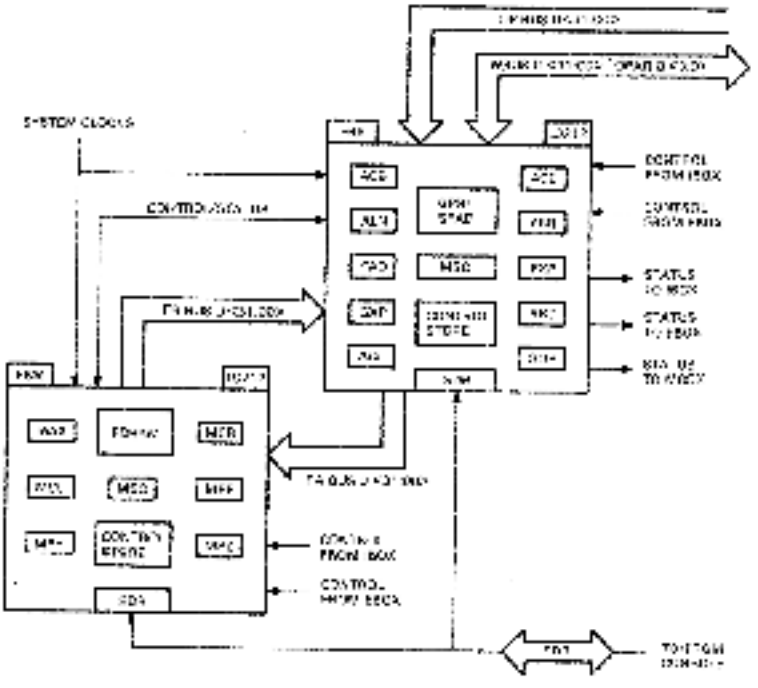


Figure 12-1 FBox main Block Diagram

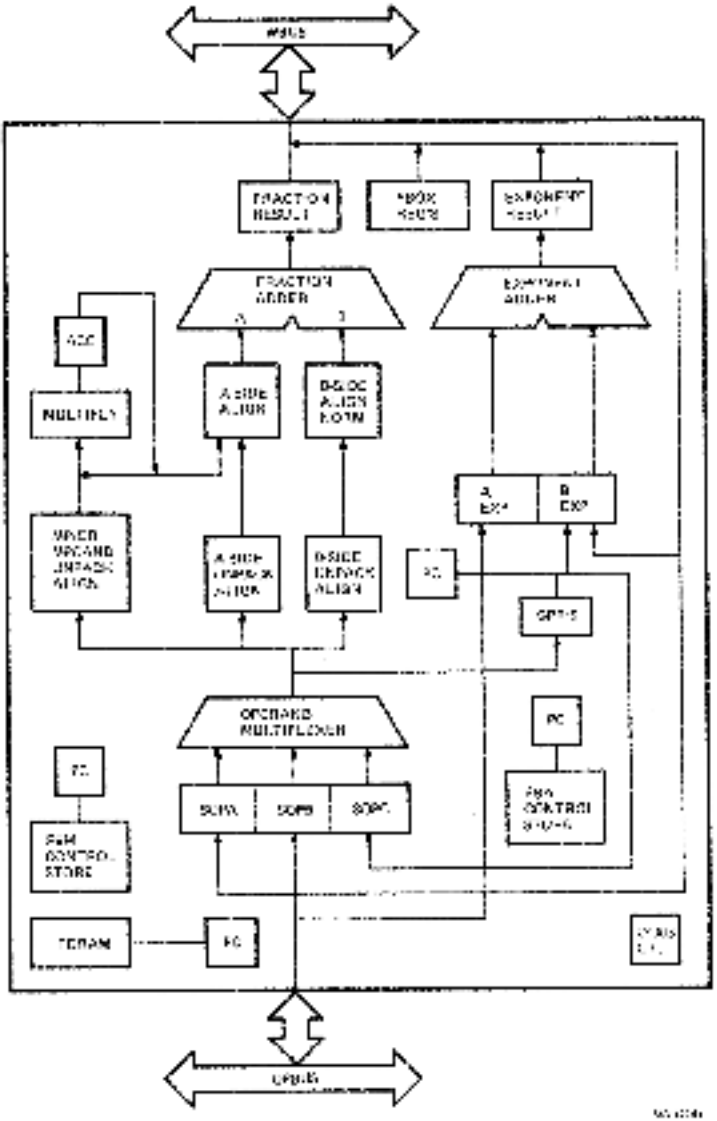


Figure 12-2 FBox Block Diagram

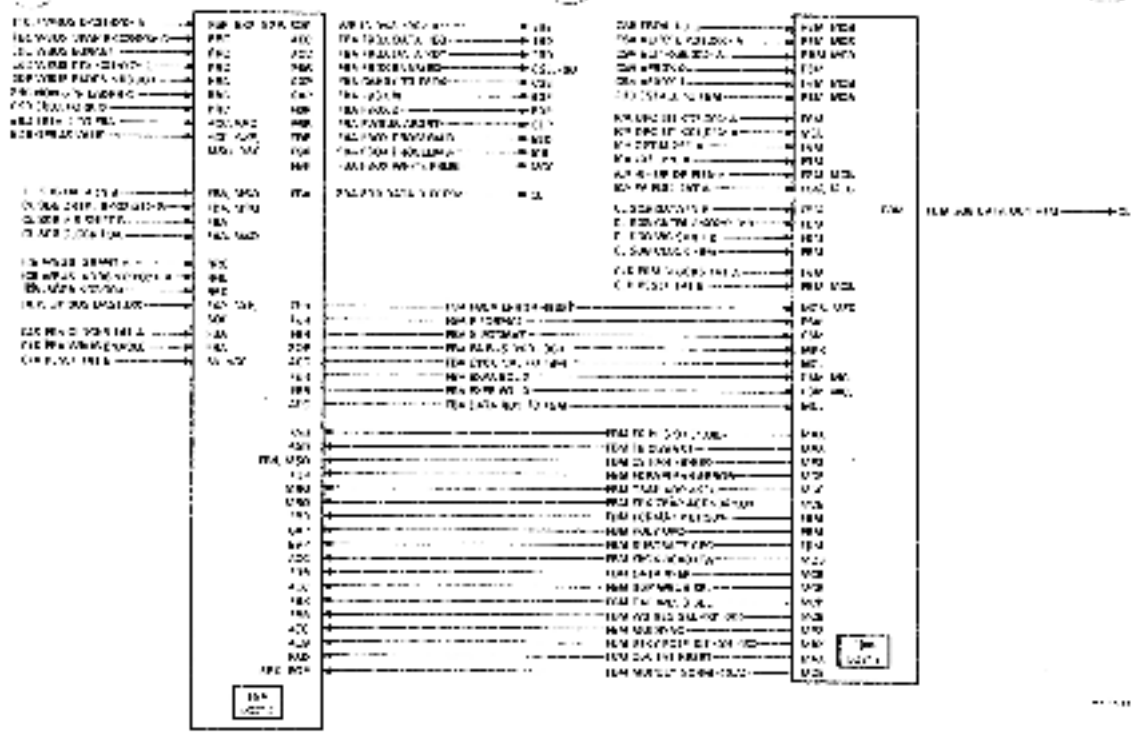


Figure 12-3 XBox Interface

12.1 FBOX MICROCODE

12.1.1 FBox Adder (FADA) Microcode

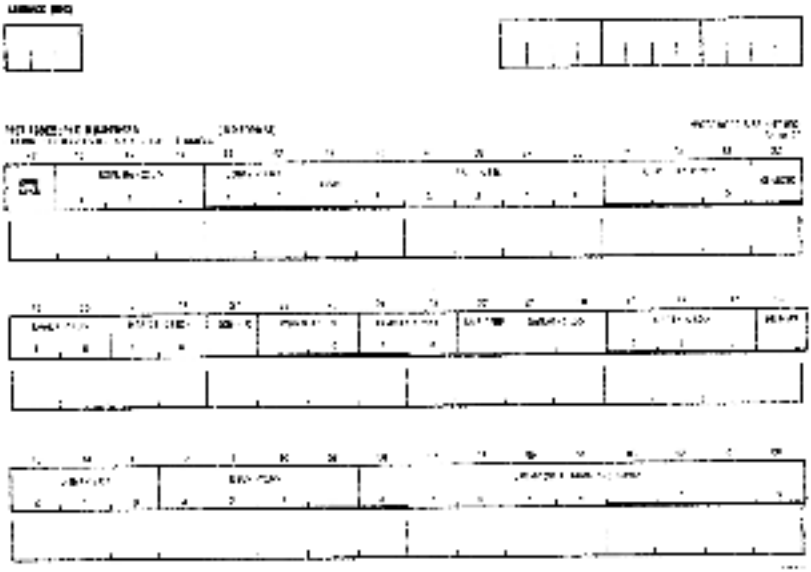


Figure 12-4 FBox Adder Microcode Worksheet

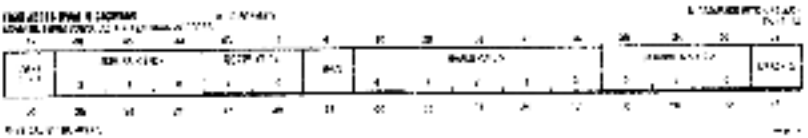


Figure 12-5 FADA Microword <47:32>

DWBT SPAD<0>

This bit specifies reading or writing the scratchpad, and has the following encoding:

0 = read scratchpad
1 = write scratchpad

OBSL RA<2:0>

This field selects the scratchpad or GPR address, and has the following encoding:

0 = Base address (30) for scratchpad locations 30 - 3F

1 = Base address (20) for scratchpad locations 20 - 2F

2 = Base address (10) for scratchpad locations 10 - 1F

3 = Base address (00) for GPRs in locations 00 - 0F

4 = GPR/SP address = GPR/SP address -1

5 = hold the GPR/SP address

6 = GPR address from PBox for optimized instruction

7 = GPR address = previous address -1 if OPBus is valid, otherwise GPR address = previous GPR address

DDRTY<1:0>

This field controls DATA READY, DATA REQUEST, and holding the EOPC latch. The field has the following encoding:

0 = NO OP

1 = At T3 assert DATA RDY. At T1 hold EOPC

2 = At T3 predict DATA RDY for next cycle, else, if in rdmw assert DATA RDY. At T1, clock the conditions in the TDP

3 = At T3 assert DATA REQUEST

OPAR<0>

This bit reflects the microword parity, and has the following encoding:

0 = microword has odd parity, parity bit is reset
1 = microword has even parity, parity bit is set

OBAZU<4:0>

This field controls the PXP, CXP, and PBX. The following field encoding specifies the selected macro def n items:

00 = COMP.DIGK	10 = SPARE 4
01 = CXP BIT	11 = SVAD ADPA - XOPB
02 = SPARE 3	12 = ADISN ADD CXP
03 = PXP DIFF	13 = SPARE 2
04 = LOAD EXP	14 = CLOCK SKIP TEST ERROR
05 = MBOX -> XOPA	15 = SET DIVIDE BY 0
06 = MBUS = 1	16 = SUM AND LOAD BIAS
07 = CLR XAP	17 = SUB AND LOAD BIAS
08 = LD BYT MSKR	18 = DIVIDE ADJ EXPONENTS
09 = CLR XOPB	19 = MULTADJ EXPONENTS
0A = CLR XOPA	1A = ADJ ADD EXP
0B = HOLD EXPR	1B = SPARE 1
0C = MBUS -> XOPB	1C = SAVE EXPB
0D = LOAD 32	1D = SUBP ADPA - XOPB
0E = LD BIT MSKR	1E = CLR EXCEPTS
0F = LD SD -> XOPB	1F = NORM EXPB

CFMSEQWRK<2:0>

This field specifies loading of the latch that controls the bit rotator. The field has the following encoding:

0 = DIVIDR, set up for a rotate

1 = HOLD, do not change the content of the BIT REV latches

2 = UNDEFINED, not used

3 = UNDEFINED, not used

4 = NO ROTATE, do not rotate

5 = NORM, load bit normalize count

6 = ALIGN, load bit alignment count

7 = DIVIDR UNPACK, load F (F or D - format) or G (G-format) for divide unpack

FPADDIG<0>

This field defines the operand significances for addition or subtraction. The field has the following encodings:

0 = data being processed by fraction adder is of low significance

1 = data being processed by fraction adder is of high significance



Figure 12-6 FPA Microword <31:16>

FPADDCTL<1:0>

This field controls the fraction adder, and has the following encoding:

0 = ADD, add A + B

1 = CONDB, add or subtract A+B depending on op code and op code sign

2 = DIV, add or subtract A±B depending on previous quotient bit

3 = SUBT, subtract A - B

LMSTOR<1:0>

This field controls the WCSA, and has the following encoding:

0 = load PAD, clear AXT

1 = load BYT WDR, ROT and MSPX potentially unpacked, aligned, or normalized

2 = XBUS, clear

3 = HOLD

CSELPSX<0>

This field controls loading the PARA and PNBE and the input to the SHFTX MUX, and has the following macro definition encoding:

0 = READ.PNBS
1 = LOAD.ASDI

CSOURX<1:0>

This field selects the inputs to the source operand muxes, and has the following macro definition encoding:

0 = PARA.OPMUX
1 = MDCS.PARA
2 = OPBUS.OPMUX
3 = CPRRCS.OPBUS

OPARAM<1:0>

This field controls loading of the PARA, and has the following macro definition encoding:

0 = LOAD.F.G, load the packed high bits of G-format; load PARA <31:00>. GROVER will rotate PARA output left 4 bits.

1 = LOAD.H.DF, load the packed high bits of F/D-format; load PARA <31:00>. GROVER will rotate PARA output left 3 bits.

2 = LOAD.L.C, if the previous PARA was 0, load the packed low bits of G-format, load PARA<31:00>. GROVER will rotate the PARA output left 4 bits.

LOAD.L.DF, if the previous PARA was 1, load the packed low bits of F/D-format, load PARA<31:00> GROVER will rotate PARA output left 8 bits.

LOAD.NOROT, if the previous PARA was 2, load PARA <31:00>. GROVER will rotate PARA output left 3 bits.

3 = HOLD, hold PARA.

FBOX
Fixed Adder (FBA) Microcode

OALNSIC<00>

This field determines the significance of operands for alignment, and has the following encoding:

0 = process low-order bits
1 = process high-order bits

OAUXX<1:0>

This field CONTROLS THE AUXA and AUXB latches, and has the following macro definition encoding:

0 = NOP
1 = LOAD.AUXA
2 = LOAD.AUXB
3 = LOAD.AUXA.AUXB

OBOIX<2:0>

This field controls the AMX/BMX inputs and the BSIDE byte rotator, and has the following macro definition encoding:

0 = INPUT.A.B
1 = INPUT.B.A
2 = BLA.FD.CKT
3 = ALN HOLE.CKT
4 = DATA.A.D
5 = DATA.B.A
6 = DATA.A.D.XBN
7 = NORM

UEHMX<0>

This bit generates the C input to the AMX, and has the following macro definition encoding:

0 = K.ZERO, generate zero for AMUX.K

1 = K.RND.DT, generate a rounding constant dependent on the data type for AMUX.K

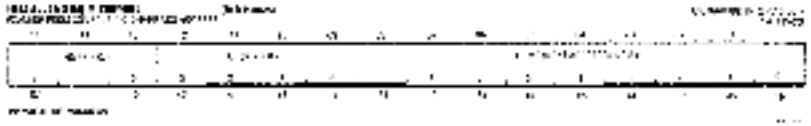


Figure 12-7 PBA Microword <15:00>

UENX<2:0>

This field enables the hidden-bit MUXs, and has the following macro definition encodings:

0 = KLUDGE
1 = S.ADAD
2 = DEPT.C
3 = BSDC.NXTBY
4 = AUXA.PBBY
5 = ADAL.AUXB
6 = AUXA.C
7 = ADAL.NXTLY

UBBK<3:0>

This field enables and/or disables the UJUMP field, and has the following macro definition encoding:

0 = IxD and DATA IN
1 = PDR.DIST.KCPM.BCTP
2 = BD.INPO
3 = not used
4 = PAD.BBBO
5 = MUL.SYNC
6 = DATA SYNC
7 = DATA.IN
8 = EXP.BBBO
9 = CPYA1 and EXP.DIFF
A = not used
B = EXP.DIFF <04
C = CRC.BB
D = PITUPN
E = CALL
F = NOP

UJUMP<8:0>

This field generally defines the next micro address, but may be modified by the UBBK field which is ORed into it.

12.1.2 EBox Multiplier (EBM) Microcode

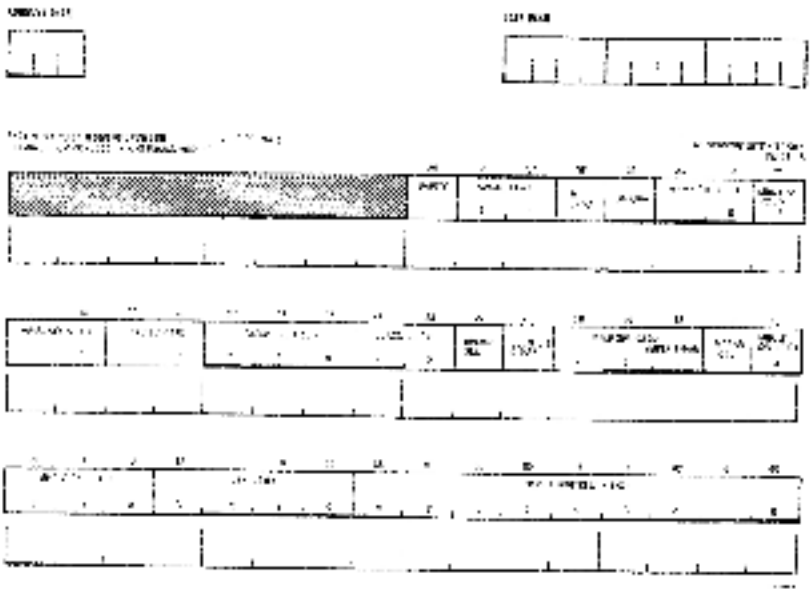


Figure 12-8 EBox Multiplier Microcode Worksheet

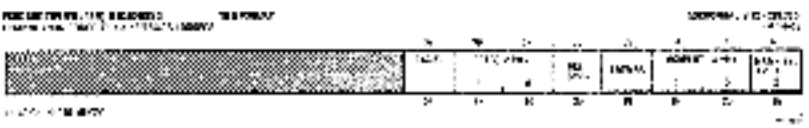


Figure 12-9 PBM Microword <39:32>

PARITY<0>

This bit selects the microword parity, and has the following encoding:

    0 = ODD, uword has odd parity and parity bit is reset
    1 = EVEN, uword has even parity and parity bit must be set

SPARE<1:0>

This is a spare field and all encoding is 0 and is decoded as NOP.

MUL SYNC<0>

This bit informs the PBA that the PBM has result data ready, and has the following encoding:

    0 = NOP
    1 = SYNC, send SYNC to PBA

LDMSKACC>

This bit enables loading of the PMRA register, and has the following encoding:

    0 = HOLD PMRA
    1 = LOAD PMRA

MSMX CTL<1:0>

This field selects the input to the MSMX mixer, and has the following macro definition encoding:

    0 = EXACC
    1 = EXACC and 0
    2 = EXACC and BYPASS
    3 = PMRA and BYPASS



Figure 12-10 FBM Microword <31:16>

MBMX SEL<2:0>

This field selects the input to the MBMX mixer, and has the following encoding:

0 = PROD.F MBMX enabled for FMUA unshifted
DIVISOR.LOW

1 = PROD.D.HIGH MBMX enabled for FMRA unshifted
QUOT.D.HIGH

2 = PROD.G.HIGH FBMX enabled for FMRA shifted left 1 bit
QUOT.D.HIGH

3 = not used
4 = QUOT.F MBMX enabled for BXACC unshifted
ACC.LEFT24

5 = PROD.D.LOW MBMX enabled for BXACC unshifted
QUOT.G.LOW

6 = PROD.G.LOW MBMX enabled for BXACC shifted left 1 bit
QUOT.G.LOW

7 = ZERO MBMX disabled and output is 0

BXAC LD<1:0>

This field enables loading of the BXACC latches, and has the following macro definition encoding:

0 = NO.Q
1 = MBMX.TO.BXACC
2 = Q.TO.BXACC
3 = NOP

CARRY CTL<2:0>

This field controls the Carry Save logic, and has the following encoding:

0 = add ACC CLACOUT A and ACC CLACOUT B to the partial product

1 = add ACC CLACOUT B and ACC CLACOUT C to the partial product

2 = add ACC CLACOUT C and ACC CLACOUT D to the partial product

3 = add ACC CLACOUT D to the partial product

4 = load CRY A

5 = load CRY B

6 = load CRY C

7 = not used

LDACC<1:0>

This field controls the loading of the ACC, and has the following encoding:

0 = hold accumulator
1 = shift ACC right 8 bits
2 = shift ACC left 24 bits
3 = clear accumulator

QUMUX SEL<0>

This bit selects the input to the XUQX multiplexer (multiplicand or quotient), and has the following encodings

0 = SEL.MCAND, select XJQX for MCAND
1 = SEL.QUOT, select XUQX for quotient bits

COUNTER CLEAR<0>

This bit examines and clears the Quotient Bit Counter, and has the following encoding:

0 = INC, enable quotient bit counter to increment
1 = CLR, initialize quotient bit counter to C0C1

MUPP SEL<2:0>

This bit selects the active 8-bit multiplier, and has the following macro definition encoding:

0 = MMLUC.LC, select MMLUC byte 0
1 = MMLDD.BC, select MMLDD byte 0
2 = MMLDC.LI, select MMLDC byte 1
3 = MMLDD.BI, select MMLDD byte 1
4 = MMLUC.UI, select MMLUC byte 2
5 = MMLDD.BI, select MMLDD byte 2
6 = MMLUC.UI, select MMLUC byte 3
7 = MMLDD.BI, select MMLDD byte 3

MCAND SEL<0>

This bit selects the active 32-bit multiplicand, and has the following encoding:

0 = SEL.XMLDB, select MCAND MUX for XMLDA
1 = SEL.XMLDB, select MCAND MUX for XBLDB

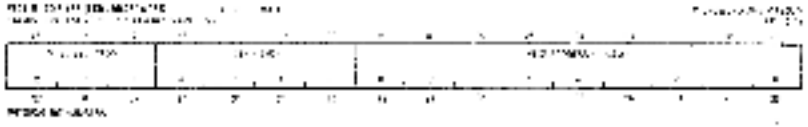


Figure 12-11 FPM Microword <15:00>

MHLD <1:0>

This field controls the loading of the multiply hold latches, and has the following macro definition encoding:

```
0 = LOAD.A,F
1 = LOAD.A,U,N
2 = LOAD.A,C,B
3 = LOAD.A
4 = SPARE
5 = DATA.LOAD.A,C,I
6 = LOAD.AB,C,L
7 = LOAD.B
8 = LOAD.C,F
9 = LOAD.C,U,N
A = LOAD.C,C,E
B = LOAD.C
C = HOLD
D = LOAD.CD,D,L
E = LOAD.CD,C,L
F = LOAD.D
```

BRN<3:0>

The branch double field allows the modification of the NEXT ADDRESS (JUMP) field, and has the following macro definition encodings:

```
0 = OPCODE.VAL and TRD
1 = not used
2 = DATA.SYNC
3 = SEQUENCER.YIELD
4 = not used
5 = not used
6 = not used
7 = not used
8 = OPCODE.VALID
9 = not used
A = not used
B = not used
C = not used
D = RETURN
E = CALL
F = MAP
```

NEXT ADDRESS<8:0>

The NEXT field specifies the next microaddress. The field may be modified by the contents of the BRN field which is ORed into it.

12.1.3 FBox Dispatch RAM (FDRAM)

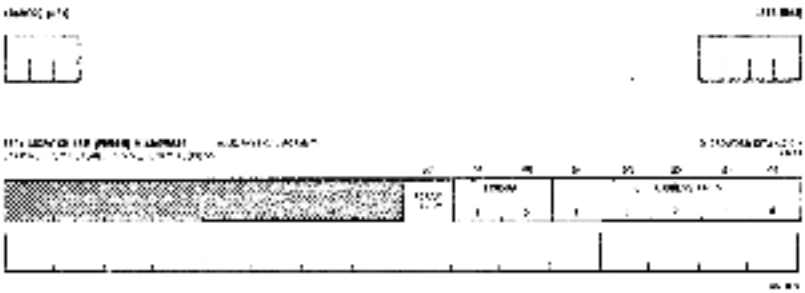


Figure 12-12 FBox Dispatch Ram (FDRAM) Microword

NOTE

The GLD and physical bit number assignments are identical.

FDRAM PARITY<0>

This bit represents the odd FDRAM parity.

FDRAM<0>

This field is enabled only when an OP Code is being decoded. The field is routed to the FER where it is decoded to provide G-Format, F-Format, and I-Format (Integer) data formats.

FDRAM ADDRESS<9:0>

This field is routed to the FRA and FRB FSMs as the micro PC for fork and trap conditions.

12.1.4 FBox Substitution Modules

If the FBox is removed for troubleshooting, or any other reason, the FBox substitution modules must be installed. The FBox Jumper Module (FJM), an LO218, will replace the FRM (LO213) in slot AC7. The FBox Terminator Module (FTM), and LO223, will replace the FBA (LO212) in slot AC8. The FBox terminates the MBus and OPBus.

12.1.5 Turning the FBox ON and OFF

The FBox can be turned on and off via the Accelerator Control and Status register (ACCS - IPR number 28). This register may be written or read via any Macro program or alternatively via console commands. When the FBox is turned off, EBox microcode will handle those floating point instructions normally executed by the FBox.

The default mode of operation is for the console to enable the FBox (if present) upon booting. VMS, in turn, will also attempt to turn on the FBox at system startup time. Note that in order to disable the FBox while running VMS, you must write the ACCS register AFTER VMS has been booted.

12.1.5.1 To Disable the FBox - After the system is up and running a Macro program (such as VMS):

```
^P                      ; enter CIO mode
>>>HALT                 ; halt the processor
>>>DEPOSIT ACCS 0       ; turn the FBox off
>>>CONTINUE             ; then continue
```

12.1.5.2 To Enable the FBox - After the system is up and running a Macro program (such as VMS):

```
^P                      ; enter CIO mode
>>>HALT                 ; halt the processor
>>>DEPOSIT ACCS 3000    ; turn on the FBox
>>>CONTINUE             ; then continue
```

12.1.5.3 To Determine the Status of the FBox - After the system is up and running a Macro program (such as VMS):

```
^P                  ; enter CIO mode
>>>HALT             ; halt the processor
>>>EXAMINE ACCS     ; read the ACCS register
    I 28 C0000001   ; ACCS contents printed on console, where
                        X=0 if FBox is disabled
                        X=8 if FBox is enabled
                    ; Note that the '1' digit refers to the
                    ; accelerator type and is set by microcode
                    ; Refer to the register description for ACCS
>>>CONTINUE         ; then continue
```

NOTE

There is a console flag called 'FBOX' which may be set or cleared via the SET FBOX ON and SET FBOX OFF commands. In addition, the status of this flag will be displayed with the SHOW FLAGS console command.

This command DOES NOT directly control the enabling or disabling of the PBox. Its purpose is to control the action of the INIT console command regarding the enabling or disabling of the PBox. If set ON, the PBox (if present) will be enabled by the INIT, INIT/UNI, or INIT/PARM commands.

Accessing the ACCS register is the only way to determine the actual status of the PBox and to control enabling and disabling it.

CHAPTER 1:

INDEX

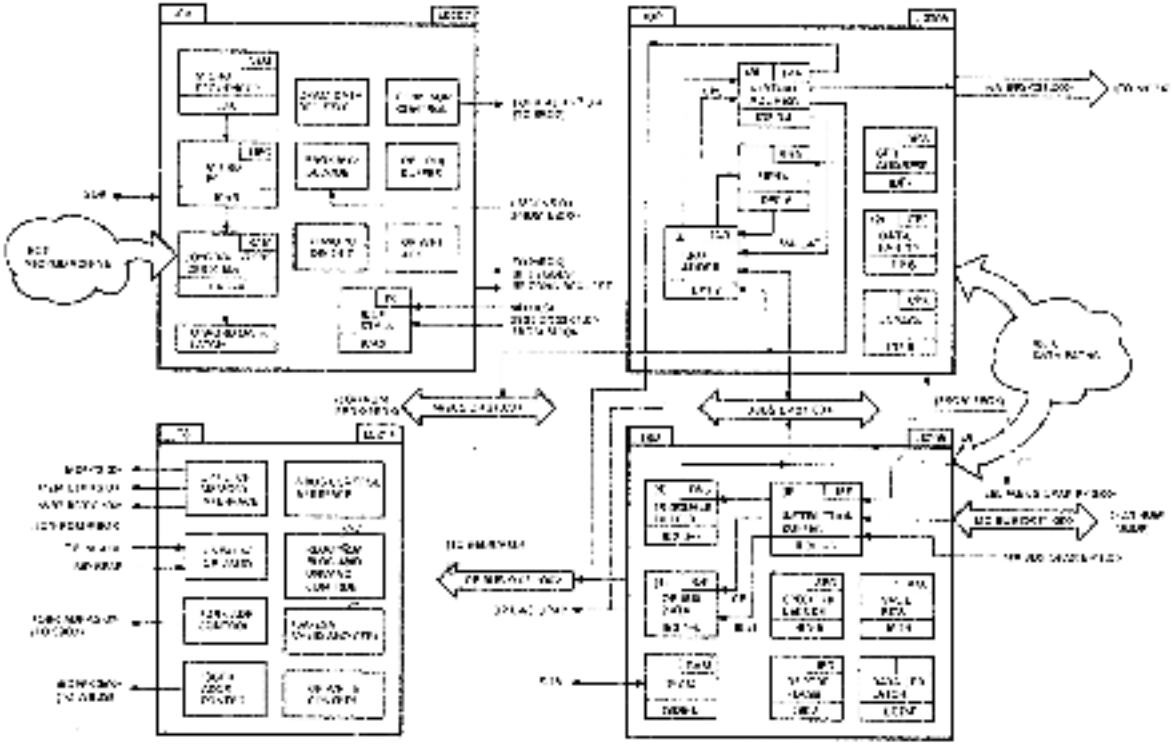


Figure 13-1 IBox Module Block Diagram

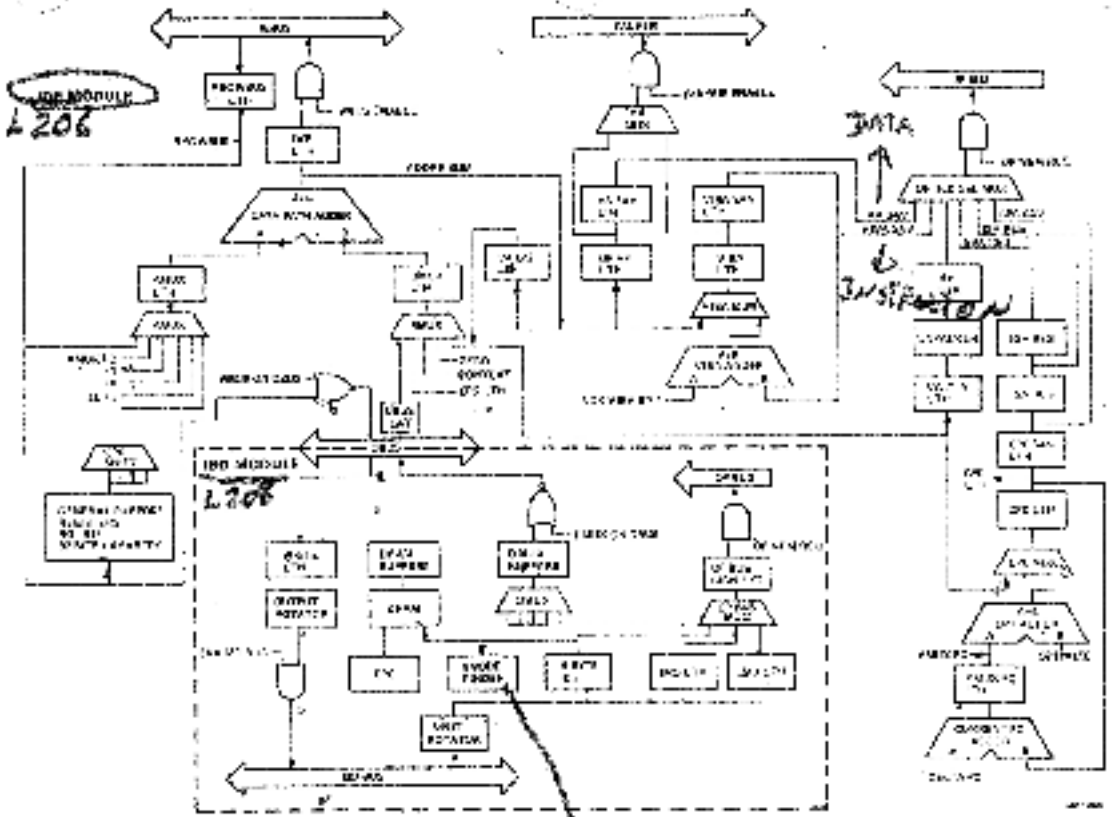


Figure 13-2 IBox Data Paths Block Diagram

13.2 IBOX MICROCODE

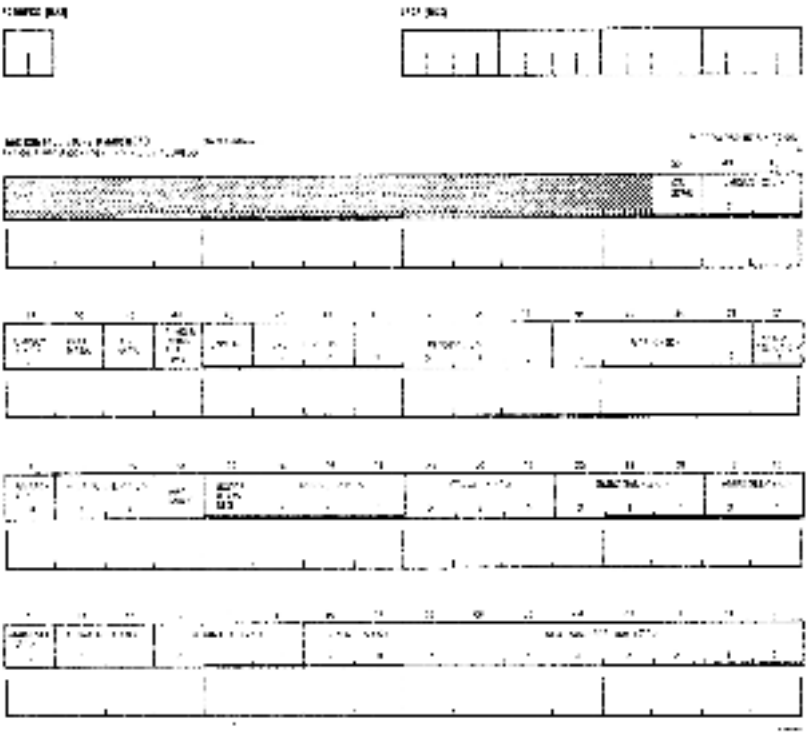


Figure 13-3 IBox Microword Worksheet



Figure 13-4 IBox Microword <50:48>

TCS OPA<0>

The MICRO PARITY field contains odd parity calculated on the uword.

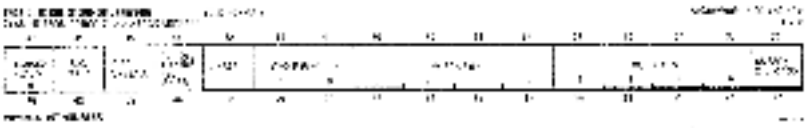


Figure 13-3 IBox Microword <67:32>

UNISC2<2:0>

The MICRO MISCELLANEOUS field provides those miscellaneous functions which do not require the assignment of an entire field for control.

```
0 = NOP
1 = INSTR.IBG
2 = IBS.OPHIT.FAULTY
3 = UNUSED
4 = FORCE.VRT.SYS.VALID
5 = IBUF.PLSEL.SB.VALID
6 = FORCE.VRT.SYS.ENA.OPHIT.PARITY
7 = EXTRA.FIX
```

IBOX MARK<0>

The IBOX MARK field is used to mark a microinstruction for microcode or hardware debugging.

```
0 = Normal:  no processing

1 = Microbreak
```

DIAG UNSTALL<0>

The DIAGNOSTIC UNSTALL field is used only in diagnostic mode, and will override all IBox stall conditions.

```
0 = NOP
1 = Override any IBox stall condition.
```

IB HOLD (INHIBIT IBOX SHIFT)<0>

This field inhibits the IBOX from shifting, and is used only in diagnostic mode.

```
0 = Nohold
1 = Hold
```

UNSTALL (DIAGNOSTIC UNSTALL)<0>

This field is used in some IBox microtrap routines to override all IBSTALL cases: IBSTALL due to an MTPR IBUF command, or due to DRAM SCRP.

```
0 = NORMAL
1 = Override STALL
```

CYCLE ID<1:0>

The CYCLE ID field classifies the current cycle into one of the following types:

0 = NORMAL: NON-IFORK CYCLE
1 = PROBE IFORK ENTRY (RX)
2 = BASE OPERAND ADDRESS (BOA) ENTRY
3 = IFORK ENTRY AND =(BOA+RX).

MISC<1:0>

The MICRO MISCELLANEOUS field handles miscellaneous IBox functions.

0 = NOP
1 = Spill write Fbox GP.MPIE command
2 = FB.FLUSH.LD.CPC
3 = IB.EI.RH
4 = CON.BRANCH
5 = IB.FLUSH.ICCND
6 = READ.TMP
7 = ENT.SH.IFORK
8 = Reserved Addressing Mode (RAF)
9 = DIAG.DONE
A = DIAG.RSSBI
B = UNLK.CHK.IFX
C = POP Stack
D = Clear CPU VALIDs for Flush and Branch
E = COND.FA.DP.MEM.REQ
F = FA.GP.PTM.PTC

MCP<3:0>

The MICRO MEMORY CONTROL FUNCTION field defines the DP PORT MCP field.

0 = READ.RLOK if DRAM MEM equals read and READ.WCHK if DRAM MEM equals modify
1 = NO DP-PORT MEMORY REQUEST
2 = WRITE.V.NOPAGE
3 = WRITE.V.NOPAGE.2ND
4 = UNUSED
5 = WRITE.V.WCHK
6 = UNUSED
7 = UNUSED
8 = READ.V.NORM
9 = UNUSED
A = READ.V.WCHK
B = READ.V.RCBK.2ND
C = READ.V.NOPAGE
D = READ.V.NOPAGE.2ND
E = IB.FILL.GP
F = IB.FILL.IBP

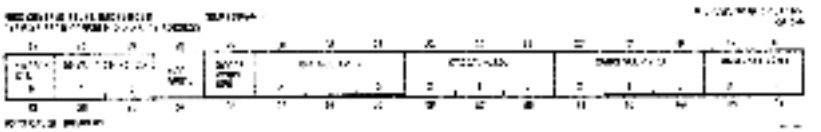


Figure 13-6 IBox Microword <31:16>

UNPACK CTL<1:0>

The UNPACKER CONTROL field determines how the IB register data should be unpacked before being driven onto the OP Bus.

0 = NOP
1 = SIGN EXTEND IF -ASBI
2 = Unpack as short literal according to DRAM CTL and type fields.
3 = Unused

OP VALID CTL<1:0>

The OPERAND VALID CONTROL field controls the setting of the OP VALID flag for the IB and IMD operand buffers.

0 = NOP
1 = Set OPVALID if (!READ + MODIFY + VMSC)* RMODE) + -(ASBU * SMODE)
2 = Set OPVALID if ASBC + (!READ + MODIFY) * ALIGNED)
3 = Set OPVALID unconditionally

REG MODE<0>

The REGISTER MODE field marks the beginning of register mode specifier processing. It determines the validity of an entry into the scoreboard. These are the only cases when scoreboard entries are validated.

0 = Not an RN specifier
1 = RN specifier

UCODE WBUS REQ<0>

The UCODE WBUS REQUEST bit partially controls updating the RLOG and requesting an IBox WBus cycle.

0 = RLOG entry is pushed if this is the first IFORK for this OPCODE. WBus is conditionally requested if doing an CNMIRD (GPR SPC. = 3)
1 = RLOG entry is pushed with valid GPR number and context entry. WBus is requested.

IBOX
IBOX MICROCODE

GPR SRC<2:0>

The GPR SRC<2:0> field determines the source of the GPR read address.

0 = GPR address from IBOX <31>
1 = Index register is being addressed by a saved index register number
2 = Last GPR address plus 1
3 = GPR address in previous GPR address
4 = ALOG unwind
5 = GPR ADDRESS <31:0> from CTX.CTL<10>; GPR address <11:6> from UNWALK.CTL<11:0>
6 = UNUSED
7 = UNUSED

CTX CTL<2:0>

The CONTEXT CONTROL field controls the various contexts (data types) for the adder, memory references, and RLOG writes during a microcycle.

CONTEXTS (IN BYTES)

	ADDER CTX	MEM CTX	RLOG CTX
0	DRAM CTX	DRAM CTX	IBAX CTX
1	0	4	4
2	X	2	X
3	= BLOG CTX	X	X
4	-DRAM CTX	DRAM CTX	-IBAX CTX
5	-4	4	X

X = UNDEFINED

BMUX SRC<2:0>

The BMUX select field controls the input to the BMUX.

0 = ZERO
1 = VA
2 = Adder CTX as defined by CTX CTL
3 = UNUSED
4 = PMX.TMP
5 = LMX.IMD
6 = CPC, used for serial command flushes
7 = Hold VA, inhibit VA clocking. Not a BMUX SEL function



Figure 13-7 IBox Microword (15:00)

ANUX SSL(2:0)

The ANUX SELECT field controls the input to the ANUX.

```
0  =  PCPD
1  =  ANDA .NE.
2  =  +4
3  =  -4
4  =  OPP, if MBus match then replace GPR data with MBus data
5  =  OPP (Left Shift 1)
6  =  OPP (Left Shift 2)
7  =  MBUS
1-0 =  Enable ISTALL if scoreboard bit
2-6 =  ENABLE ISTALL if MBus match
```

MICRAP CTL(1:0)

The MICROTRAP CONTROL field enables a microtrap to occur in the next cycle if the current OP PORT memory request is unaligned.

```
0 = Inhibit Microtrap due to unalignment
1 = Micro-trap last cycle for unaligned instruction fetch
2 = Micro-trap for unaligned (Rx) or any other unaligned
    operand
```

IPORX CTL(1:0)

The IPORX CONTROL field determines if the next cycle is an IPORX entry or not. The field has the following encoding (where B = byte, W = word, L = longword):

```
0 = No new IPORX
1 = IPORX IF A200 = (READ' (BWL))
2 = (IPORX IF VSPC & WRITE = (READ - SOJGTZ) ' (BWL))
3 = IPORX IF BSAE ^ (BWL)
4 = IPORX IF CORR
5 = IPORX IF ESOX JMIP <counts 24
6 = Unconditional IPORX
7 = unused
```

(BRNM<3:0>

The BRANCH MNEMONIC field enables a multi-way (up to 16-way) branch in microcode in the absence of stack or microtrap.

IDRAN MEM BITS

	3	2	1	0
0	0	0	0	0
1	0	QUAD 4 OPTS	IDRAN MEM 1	IDRAN MEM 0
2	DLOG FIRST	IDRAM CTX 2	IDRAM CTX 1	IDRAM CTX 0
4	0	0	0	0

MA<7:0>

The NEXT ADDRESS field addresses the next microinterruption in the absence of a branch, IFORK, or microtrap. For a branch, MA<3:0> are ORed with the branch conditions.

13.3 IBOX DISPATCH RAM (IDRAN)

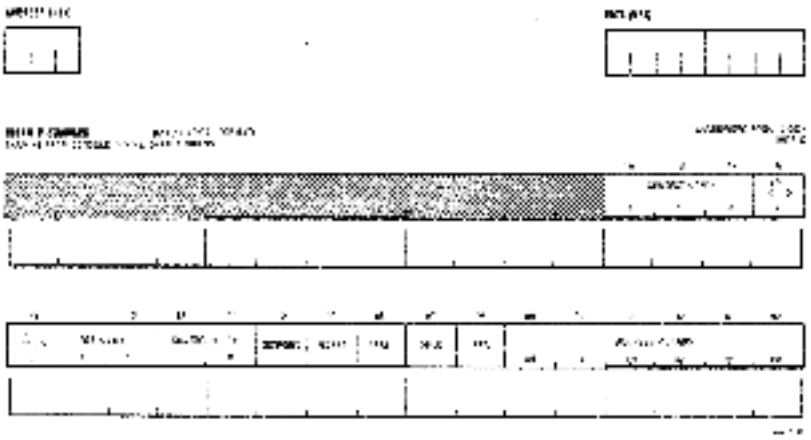


Figure 13-9 IDRAN MICROWORD WORKSHEET

13.3.1 IDRAM Address Generation

The IDRAM address is a 12-bit address which is generated from the execution point counter (EPC), the OPCODE, and a bit controlled by whether or not the instruction is extended (an FD instruction). The initial IDRAM address for any instruction has the least significant three bits equal to zero because the KAC is initially zero, being incremented for each specifier.

Table 13-1 shows how to generate the initial IDRAM address, and Example 13-1 gives examples.

NOTE

IDRAM code may be found in XADG*.MCR.

Table 13-1 IDRAM Address Generation

IDRAM ADDR	11	<13:03>	<02:00>
	FD	OPCODE <07:00>	KAC <02:00>

Example 13-1 Generating IDRAM Addresses

1. CVTLO instruction, OPCODE = 6C

IDRAM ADRS	0	0 1 1 0 1 1 0 0	0 0 0
	FD	OPCODE <07:00>=6C	KAC <02:00>

IDRAM ADRS = 00'110111000 = 370

2. CVTLH instruction, OPCODE = 6BFD

IDRAM ADRS	1	0 1 1 0 1 1 0 1	0 0 0
	FD	OPCODE <07:00>=6B	EPC <02:00>

IDRAM ADRS = 10110111000 = 5B8

13.3.6 IDRAM Microword



Figure 13-9 IDRAM Microword <15:16>

CONVERT<2:0>

The CONTEXT field specifies the data context based on the following encoding:

0 = BYTE
1 = WORD (2 bytes)
2 = LONG (4 bytes)
3 = QUAD (8 bytes)
4 = OCTA (16 bytes)
5-7 = UNPREDICTABLE

TYP<1:0>

The TYP field specifies the data types based on the following encoding:

0 = INTEGER, ASCIZ
1 = FLOATING, F, D, or G
2 = U-FLOAT
3 = VSOURCE

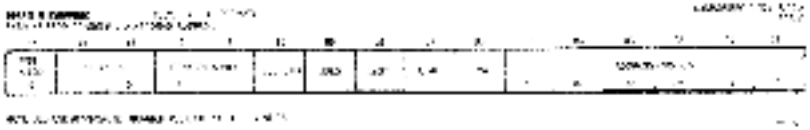


Figure 13-10 IDRAM Microword <15:00>

MOD<1:0>

The MEMORY MODIFIER field specifies the memory reference and the checking applied.

	ACCESS CHECK	
	ON READ	ON WRITE
0 = ASOURCE	-----	-----
0 = VSOURCE	-----	-----
1 = READ	READ	-----
2 = MSITE	-----	WRITE
3 = MODIFY	READ	WRITE

CONTROL<1:0>

The CONTROL (CTL) field determines the content of the FORK address. See Figure 13-1, Fork Address Generation.

0 = EXECUTE
1 = SINGLE
2 = OPT-TWO
3 = OPT-TWO & EXECUTE

CTL — FORK ADDRESS

	07	06	05	00
EXECUTE	0	DRAM(<FA>) & PEN OR -DRAM(<FA>)	DRAM ADDR <05:00>	

	07	06	05	01	00
SINGLE ^	1	DRAM(<FA>) & PEN OR -DRAM(<FA>)	DRAM ADDR <05:01>		DRAM ADDR <00> & BMODE & -HOLOUT

	07	06	05	00
OPT-TWO BMODE	-	DRAM(<FA>) & -PEN	DRAM ADDR <05> & '1' = BIGADDR	DRAM ADDR <04:00>

	07	06	05	04	03 02	00
OPT-TWO NOT BMODE	1	DRAM(<FA>) & PEN	DRAM ADDR<05> & BI-BSGMODE	DRAM TYPE< :0>	DRAM CTX<2:0>	

	07	06	05	01	00
OPT-TWO & (XBUS BMODE	0	DRAM(<FA>) & PEN	DRAM ADDR <05:01>	DRAM ADDR <00> ON UNIBUS	

	07	06	05	04	00
OPT-TWO & (XBUS NOT BMODE	0	DRAM(<FA>) & PEN	DRAM ADDR <05> & BI = '1'BMODE	DRAM ADDR <04:00>	

Where

^ If FORK ADDR bit 0 (FA<00>) = 1, turn off scoreboard

† If FORK ADDR bit 5 (FA<05>) = 1, turn off scoreboard

Figure 13-1 Fork Addresses Generation

SUSPEND<0>

If set (1), the SUSPEND (SUSP) field causes the Address Calculation Unit to suspend operation after processing the specifier that had the SUSPEND bit set.

BDSEY<0>

The BRANCH DISPLACEMENT NEXT field is set in the DRAM entry preceding a displacement. This bit is not set for Opcodes in which the first item following the Opcode is a branch displacement.

0 = Next byte after this specifier is a mode-register specifier or new Opcode.
1 = Next byte after this specifier is a branch displacement.

LAST<0>

The LAST bit indicates that the current DRAM entry is the last one for this instruction. This bit is implied if an instruction is optimized.

OPAB<0>

The ODD Parity bit is used to check the integrity of DRAM data. The console attempts to correct DRAM parity errors.

FPA<0>

The FPA OVERRIDE bit specifies that the FPA may override the RBox for the specifier that the IBox is currently processing.

ADDRESS<5:0>

The EXECUTION ADDRESS field supplies the basic FOUR address (FA) that will be sent to the EBox if no error is detected. It is modified before becoming the EBox FA.

13.4 INDEXER AND OPCODE TESTPOINTS

Table 13-2 lists oscilloscope/logic analyzer test points for the IBLPIBX (bytes 0 and 1) and Table 13-3 lists the oscilloscope/logic analyzer testpoints for the OPCODE bits (coax from the ICA to PBN) which are available on the backplane. The tables include channel number suggested for use with a logic analyzer, signal name, print page on which the signal originates, and the CPU backplane pin location.

Table 13-2 INDEXER Testpoints

Channel	Signal Name	Print	Section/Slot/Pin
$0	IBD IBUF DATA B1 7 H	IBD3	C15_00
$1	IBD IBUF DATA B1 6 H	IBD3	C15_24
$2	IBD IBUF DATA B1 5 H	IBD2	C15_12
$3	IBD IBUF DATA B1 4 H	IBD2	C15_36
$4	IBD IBUF DATA B1 3 H	IBD2	C15_44
$5	IBD IBUF DATA B1 2 H	IBD1	C15_50
$6	IBD IBUF DATA B1 1 H	IBD1	C15_60
$7	IBD IBUF DATA B1 0 H	IBD1	C15_71
$8	IBD IBUF DATA B0 7 H	IBD3	C15_02
$9	IBD IBUF DATA B0 6 H	IBD3	B15_52
$A	IBD IBUF DATA B0 5 H	IBD2	B15_56
$B	IBD IBUF DATA B0 4 H	IBD2	B15_50
$C	IBD IBUF DATA B0 3 H	IBD2	A15_08
$D	IBD IBUF DATA B0 2 H	IBD1	B15_30
$E	IBD IBUF DATA B0 1 H	IBD1	B15_10
$F	IBD IBUF DATA B0 0 H	IBD1	B15_66

Table 13-3 OP CODE Testpoints

Channel	Signal Name	Print	Section/Slot/Pin
$0	ICA OPC BIT 7 A H	ICAE	B07_83
$1	ICA EXI OPC A E	ICAE	B07_05
$2	ICA OPC BIT 7 A H	ICAE	B07_81
$3	ICA OPC BIT 6 A B	ICAE	B07_03
$4	ICA OPC BIT 5 A D	ICAD	B07_86
$5	ICA OPC BIT 4 A E	ICAE	B07_77
$6	ICA OPC BIT 3 A D	ICAD	B07_87
$7	ICA OPC BIT 2 A D	ICAD	B07_83
$8	ICA OPC BIT 1 A B	ICAB	B07_76
$9	ICA OPC BIT 0 A D	ICAD	B07_84

CHAPTER 14

INDEX

Figure 14-1 MBox Physical Organization

Figure 14-2 MBox Address Path Block Diagram

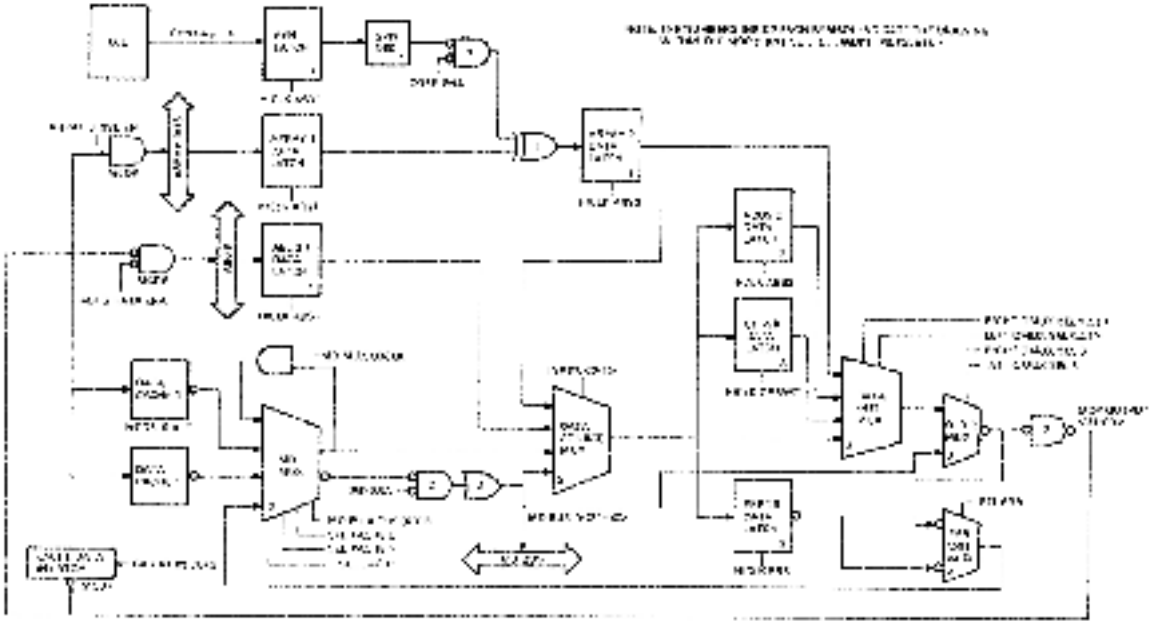


Figure 14-3 MBox Data Path Block Diagram

Figure 14-4 Mbox Interface Block Diagram

MBOX MODULES
X1015

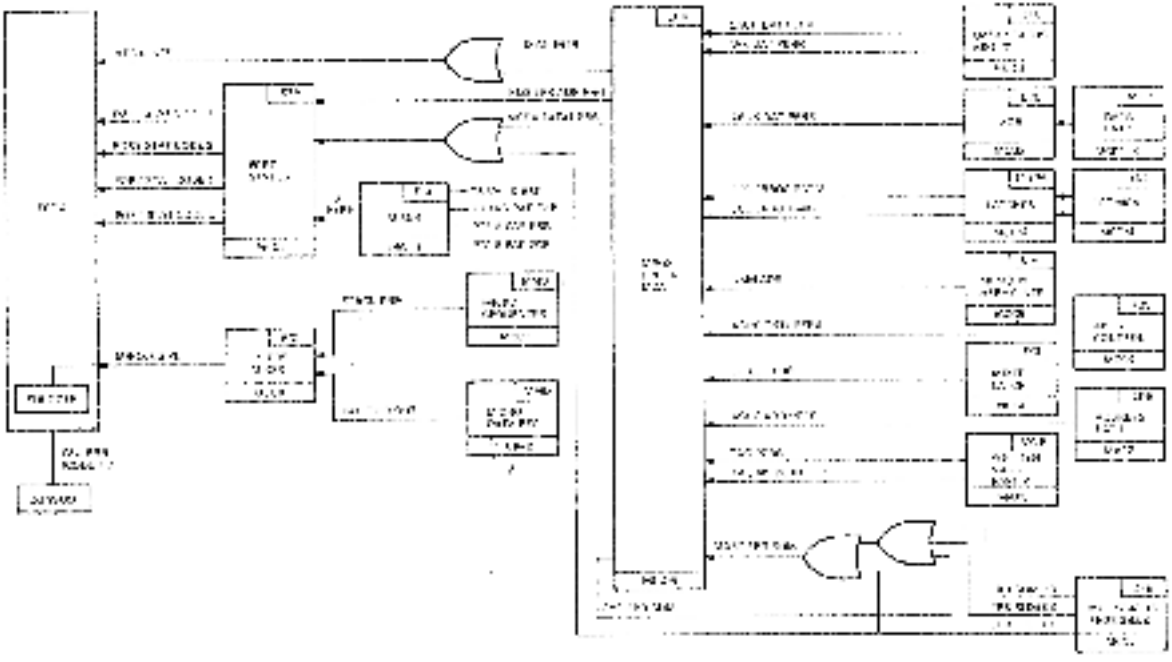


Figure 14-5 Phase Error Modeling Block Diagram

14-3 MBOX MICROCODE

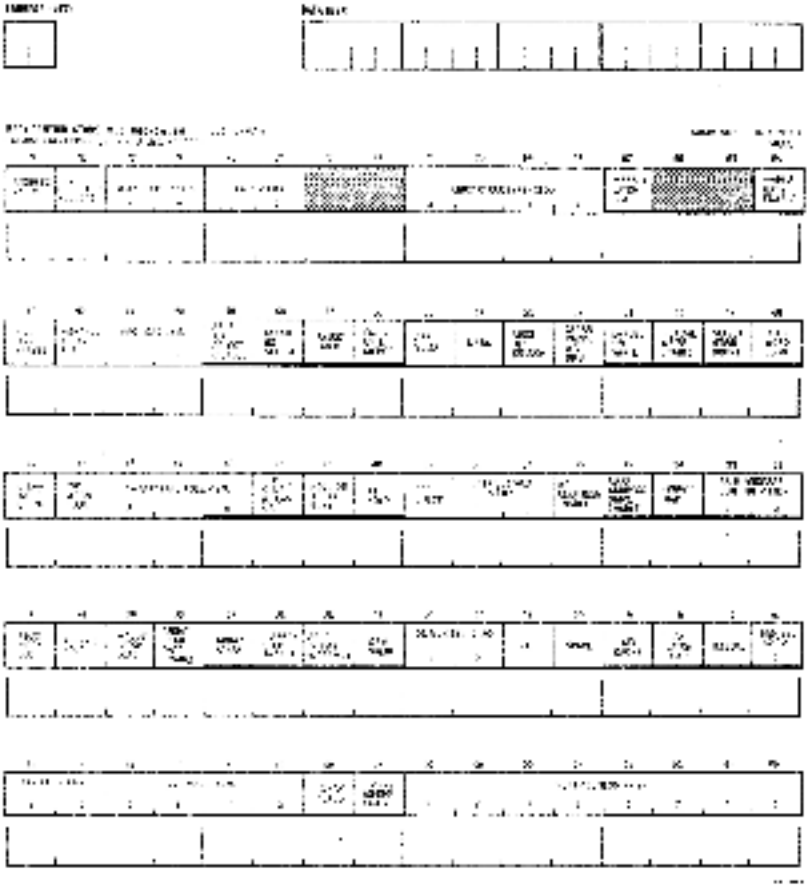


Figure 14-6 MBox Control Store Microword Worksheet



Figure 14-7 KBox Microword <79:64>

ADDRESS PARITY<0>

The address parity bit reflects odd parity of the address where the microword is stored in the control store.

RES CORR RQ<0>

When set, the RESBI CORRECTION REQUEST bit causes the data correction cycle request.

IOA SEL <1:0>

The I/O ADAPTER SELECT field provides commands to the KBus interface MCA to select the specified I/O adapter. The field has the following encoding:

0 = DMA DONE not sent to ABus
1 = LATCH B sends DMA DONE to ABus
2 = LATCH A sends DMA DONE to ABus
3 = CPA sends DMA DONE to ABus

STACK<1:0>

This field provides stack control. The field encoding is:

0 = NOP
1 = push current micro address on the stack
2 = pop a micro address from the stack
3 = when asserted, either does a return from top of stack, or return to KFUNC. ALU SEL must be specified to disable arbitration.

UNUSED FIELDS

Two 2-bit fields are unused in the KCB: <73:72> and <66:65>.

MBOX CYCLE TYPE<3:0>

Indicates the cycle type the MBox controller will perform.

```
00 = no type
01 = read register cycle
02 = write register cycle
03 = writeback cycle
04 = ASus array write cycle
05 = data correction cycle
07 = probe cycle
08 = ASus cycle
09 = CP refill cycle
0A = invalidate TB cycle
0B = IB cycle
0C = CP array write cycle
0D = CP write cycle
0E = CP read cycle
0F = refill cycle
```

-REFILL LATCH LOAD<0>

When reset (asserted low), this bit enables loading of the KCOR388 REFILL LATCH.

-ABUS DATA ENABLE<0>

When asserted low, the ABUS DATA OE bit asserts data and longword parity onto the ASus.



Figure 14-8 MBox Microword <63:48>

CB DRIVE ENABLE<0>

When asserted, CB DRIVE EN gates check bits, read from the data cache, and stored in the DATA CB MUX LATCH OUT, to the Array Bus. When there is a cache data parity error, the cache correction cycle requires that the check bits be read from the cache. The bit has the following encoding:

```
0 = do not gate CB MUX LATCH OUT to check bit bus
1 = gate CB MUX LATCH OUT to check bit bus
```

-ENABLE ARRAY BUS<0>

When asserted low, this bit enables gating 32 data bits from the CACHE WRITE DATA MUX LATCH to the Array Bus.

REG LOAD<1:0>

The REGISTER LOAD field provides inputs to the ABus interface MCS to load the I/O adapter latches, and has the following encoding:

    0 = do not load IOA latches
    1 = load LATCH A with IOA REQ
    2 = load LATCH B from LATCH A
    3 = load the latch from DATA

ABUS IOA SELECT DISABLE<0>

When set, the ABUS IOA SEL bit bit inhibits IOA SELECT if KDBX PA CONDITION is not true.

CURAR OR OFFSET<0>

The CUR OR SET bit is used for array select information for current and pending cells. If set, this bit allows loading of the current latch from the pending latch.

ABUS2 HOLD<0>

When set during DMA writes, this bit causes data to be held in the A2 DAT REG. The bit has the following encoding:

    0 = do not hold A2 DAT REG
    1 = hold A2 DAT REG

ENABLE BYTE MERGE<0>

The EN BYT MERGE bit enables the DATA OUT MUX select hardware during a CP byte write. When the DO MUX selection is for the DATA SOURCE MUX and BYT MERGE is enabled, the hardware will merge the data in the CP DATA LAT with the data coming from the DATA SOURCE MUX.

ARB SELECT<0>

This bit selects the arbiter address instead of the next micro address. However, EVEICON (and, NOT STACK EMPTY) override the ARB SEL bit and disables arbitration. The bit has the following encoding:

    0 = select next micro address from arbiter
    1 = select next micro address from NEXT ADDRESS field

MARK<0>

This bit is a trace bit used for debugging.

ENABLE REFILL BRANCH<0>

When asserted low, this bit enables the optimized refill branch logic.

CLEAR CACHE V/V BITS<0>

When asserted with the CACHE WRITE ENABLE bit, this bit will cause the written word valid (V/V) bits to be set to 0.

ENABLE CPU WRITE<0>

This bit enables CPU to be written, switching cache selection.

<CACHE WRITE ENABLE<0>

When this bit is asserted low, it enables writing the cache data and tags.

SELECT WORD COUNT<0>

When a refill, writeback, or XBox multiword array reference takes place, the word counter keeps track of the current longword being read or written. IF SELECT WORD COUNT is set the word counter is selected for addressing the cache. It reset, the physical address is used.

LOAD WORD COUNT<0>

When asserted, the LD WD CNT bit will load the longword counter from PA LATCH <03:02>.



Figure 14-9 MBox Microword <47:32>

CLEAR WORD COUNT<0>

When asserted, the CLR WD CNT bit will clear the longword counter in the WRP MCA on the address module.

INC WORD COUNT<0>

The INC WD CNT bit increments the longword counter.

NBOX
NBOX MICROCODE

PA MUX LATCH SEL<2:0>

This field controls the PA MUX on the Address Path module, and determines the input to the PA MUX LATCH. The field encoding is as follows:

0 = PA MUX
1 = Refill Latch
2 = Write Latch
3 = ABus Latch
4 = TB PTE Store
5 = TB TAG Store
6 = CH TAG Store
7 = CI TAG Store

-ABUS CLEAN UP<0>

The ABUS CLEAN UP bit conditions an Array step. It also informs the ABus interface logic to start looking for a word count match for ABus Transfer. The bit has the following encoding:

0 = Array Step Enable causes array step
1 = ON and NOT ABUS REFILL CYCLE TYPE. (Array Step En causes array step only if NOT refill in progress AND NOT block bit.)

HOLD ON ARRAY BUSY<0>

The combination of this bit set and ARRAY BUSY true, will prevent the micro address from changing until ARRAY BUSY is false. The bit has the following encoding:

0 = do not hold here and wait for ARRAY BUSY to drop
1 = hold here until ARRAY BUSY is false

AI HOLD<0>

When HOLD is set, the AI DATA LATCH will be loaded. If reset, the AI DATA LATCH is held.

ECC CHECK<0>

The ECC CHECK field controls whether the ECC NCA on the data path module is checking data read from memory for errors (refill), or generating check bits on data being written to memory (writeback). The bit will be reset for a writeback and set for a refill. The bit has the following encoding:

0 = generate check bits
1 = check read data for errors

DATA OUT MUX<1:0>

The DO MUX field determines the DATA OUT MUX selections. On a CP byte write or an ABus masked write, the MODE selects the DATA SOURCE MUX and the hardware selects the bytes to be written. If the operation is a CP byte write, the hardware selects the CP DATA LATCH for valid bytes to be written. On an ABus masked write, the hardware selects the A2 DAT REG for valid bytes to be written. The field has the following encodings:

0 = select DATA SOURCE MUX
1 = select A2 DATA REGISTER
2 = select CP DATA LATCH
3 = select ARRAY 2 DATA LATCH

MD RESPONSE ENABLE<0>

When MD RES EN is asserted, it indicates to the CP port interface MCA that the current cycle has completed. This bit also indicates when to send the status code.

ABUS ADDRESS DRIVE ENABLE<0>

If set, the ABUS A<R DR EN bit enables the ABus drivers on the address module, allowing an address from the DA MUX LATCH to the ABus. This function is used for CP initiated requests.

*ENABLE ERROR TRAP<0>

When asserted low, the ENABLE ERROR TRAP bit is a signal to the micro address control to go to micro address 7F if there is an ABus address parity error. It forces a DOP of the block if an error trap is taken.

ABUS ADDRESS CONTROL<1:0>

This bit is a signal to the ABus interface MCA to allow control of the I/O adapter register file. The field has the following encoding:

0 = hold current address in I/O adapter register file
1 = increment if write
2 = increment current address in I/O adapter register file
3 = load current address in I/O adapter register file

NOTE

MCC ABUS ADDRS CTRL <1:0>, the signals on the ABus that control the register file, are not the same as the XBox microcode bits.



Figure 14-10 MBox Microword <3:116>

ABUS MBOX DATA<0>

When asserted, this bit is a signal to the Abus interface PCA to indicate that the MBox is ready to send data on the I/O adapter register file.

ARRAY 2 HOLD<0>

If ARRAY 2 HOLD is asserted, the ARY 2 DATA LAT will be loaded. If the ARY2 HLD bit is set the ARY 2 DATA LAT will hold data presented to the CACH WRIT BAT MUX LAT, thus preventing the CACH WRIT BAT MUX LAT from changing.

-ABUS LATCH LOAD<0>

When asserted low, this bit causes the ADR A LAT to be loaded at BTA (second T3S1). The bit is also used to clear ABUS ADR ER, and used as an enable into the SO PERIL logic.

ARRAY MEMU DATA ENABLE<0>

When asserted the ARRAY RD DATA EN bit will enable the array board drivers of the selected board. Data from the DC109 will be asserted on the Array Bus.

ARRAY START<0>

When this bit is asserted by the array interface MCA, it indicates that the microsequencer wants the array to initiate a write or read operation. The array interface MCA will issue the start signal to the array at the first I2 after the array becomes available.

-ARRAY STEP ENABLE<0>

When asserted low, this bit is used on array cycles, and conditions the array interface MCA to issue ARRAY DATA STAT to the array. Note that this is an enable function and not the actual step. Hardware issues the array step at T7 in the subsequent cycle.

ABUS INHIBIT INCREMENT<0>

This bit inhibits an address increment function from being transmitted to the Abus device if ABUS TRANSFER IN PROGRESS is not true.

DATA SOURCE MUX<2:0>

DSM VALID<0>

DS MUX SEL<1:0>

The DATA SOURCE MUX field consists of three subfields: DATA SOURCE MUX, DSM VALID, and DS MUX SEL. The overall field selects the data source to Low Byte parity generator or checker and clocked into the CP DATA LATCH, DATA ERROR LATCH, or A2 DAT REG, and sent to the DATA OUT MUX. When explicitly set it also tells the KBOX that the DATA SOURCE MUX is valid, that is, that parity counts. The field encodings are as follows:

DATA SOURCE MUX<2:0>:

    4 = select MBOX and valid
    5 = select KBMUX and valid
    6 = select A2 DATA DAT and valid
    7 = select A1 DATA REG and valid

DS MUX SEL<1:0>

    0 = MBOX
    1 = KBMUX
    2 = ARRAY BUS
    3 = ABUS DATA

DSM VALID<2:0>

    0 = No valid data at DATA SOURCE MUX output
    1 = valid data at DATA SOURCE MUX output

MIC PAR<0>

The MIC PAR bit reflects the odd mirrored parity. MIC PAR will be calculated after the address parity bit is calculated.

ABORT LAST WD<0>

ABORT LAST WD is used to force last word valid for DMA write optimization.

-SEL CACHE<0>

The -SEL CACHE bit specifies whether the MD MUX LAT will select the cache, or either the DATA OUT MUX or DATA ERR LAT based on the encoding of the DSM VALID and DS MUX SEL fields. The -SEL CACHE bit has the following encoding:

    0 = select cache
    1 = select DATA ERR LAT or DATA OUT MUX

MBOX
MBOX MICROCODE

-PA LATCH ENA(H<0>

When asserted low, the PA MUX LATCH will be loaded, otherwise it will be held.

FORCE ABUS TRANSFER<0>

Used during DMA writes to force ABus transfer in progress.

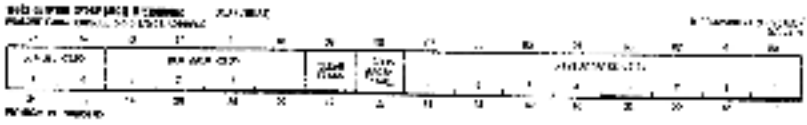


Figure 14-11 MBox Microword <15:00>

BRN SEL<2:0>

The BRANCH BRANCH SELECT field enables one of eight inputs to 8:1 multiplexers. The outputs are ORed with the four low-order bits of the NEXT ADDRESS field. Individual multiplexers are disabled through the BRN MASK field. See Table 14-1.

BRN MASK<3:0>

Each bit in this field enables one of the 8:1 branch condition multiplexers. Bit zero enables multiplexer 0, bit 1 enables multiplexer 1, and so forth. Table 14-1 lists the branch conditions that are ORed with next address <03:00>.

Table 14-1 MBox Branch Conditions

MUX SEL	MUX 3	MUX 2	MUX 1	MUX 0
0	MCT.INT. MBM	CP.BK.HIT	ABUS.REFL. REQ	W.EQ.1/MD.BK.HIT OR (NO.CACHE.HIT * LAT.RD.CYC * W.EQ.1)
1	ARRAY.BUSY	IO.WRT	REFL.PROG	CACHE.HIT
2	PAMM.SEL. ABUS	W.EQ.1. CACHE.OK	REFL.PROG	0
3	TAG.PERR	IO.MASK	BLK.HIT	IO.WRT
4	CACH.DAT. PERR	ABUS.XFR. RQ	0	PA3
5	TAG.PERR	LAST.RD	CCFP	0
6	REF.WRT	REFL.ABUS	ADR.ERR. IAT	DATA.ERR.IAT
7	ARRAY.BUSY	CP.WRT	REFL.PROC	BLK.HIT

CLEAR FLAGS<0>

When set, this bit clears the ABUS DEVICE REQUEST and CP DEVICE REQUEST flags used in MBOX address arbitration.

-ABUS MEMORY ENABLE<0>

This bit is asserted to cause a request to memory for a DMA request, and has the following encoding:

0 = enable ABus
1 = do not enable ABus

NEXT ADDRESS<7:0>

This field designates the next microinstruction address, which may be ORed with the branch conditions, or modified by the arbitration logic or stack logic.

14.3 MBOX MICRO ENTRIES

Table 14-2 is a list of the MBox micro address entries for MICRO conditions.

Table 14-2 MBox Entries on MICRO

02:	M.DIAG.READ.DOA	16:	M.CP.REFILL
06:	M.SDEP.DONE	17:	M.A-BUS.MSPILL
07:	M.DMIO.DMA.REQ	18:	M.IDLE
0A:	M.PROBE.READ	19:	M.UP.CACHE.READ
0C:	M.READ.MBOX.REG	1A:	M.CP.CACHE.HITS
0D:	M.SEND.PTE.TAG	1D:	M.PROBE.WRITE
0F:	M.READ.PCB	1C:	M.WRITE.MBOX.REG
0*:	M.SASEP.CACHE	1B:	M.CLEAR.IB.ENTRY
10:	M.WRITE.PMD	1*:	M.WRITE.PTE
12:	M.DIAG.WRITE.DOA	1F:	M.FILL.ALL.CACHE
14:	M.CACHE.READ.PAR.ERROR		

MBOX
MBOX CYCLE CONDITION CODE MICROWORD

14.4 MBOX CYCLE CONDITION CODE MICROWORD

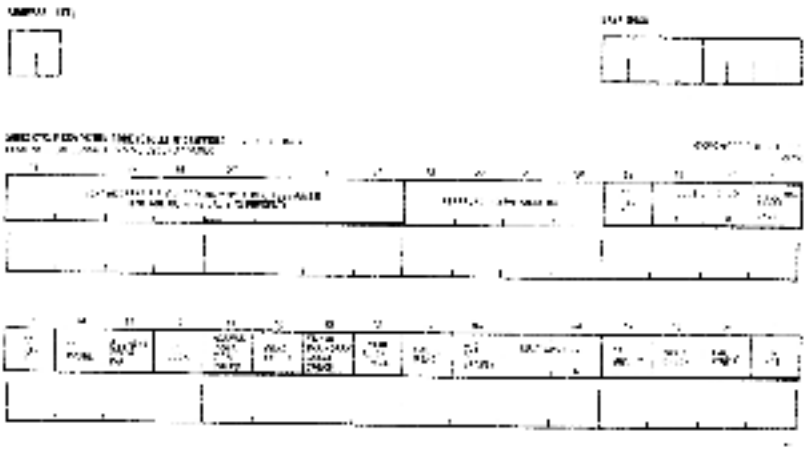


Figure 14-12 MBox Cycle Condition Code Microword Worksheet



Figure 14-13 MBox Cycle Condition Code Microword <11:1>

CP PAR 0 <30>

This bit replaces the parity calculated on the following fields:

- PID
- IPD ACC ENABLE
- DEST CP C
- CACHE READ
- ENABLE ALIGN CHECK
- CP ERROR
- ADDRESS PROT VA
- CP LOCK

-DEST CP<1:0>

This bit specifies the destination of the VA RESPONSE to the current request. The field has the following encoding:

0 = IPR
1 = GP REGS
2 = MBOX
3 = IPR FROM GP

-SECOND REFERENCE<00>

This bit is reset to indicate to the MBox logic that this is a second reference request.



Figure 14-14 MBox Cycle Condition Code Microword <15:00>

CP PAR A<00>

This bit resolves the parity calculated on the following fields:

WRITE CHECK

OPSI CP 1

3A PAGE BOUNDARY CROSS CHECK

READ CHECK

ACCESS VIOLATION CHECK

SECTION REFERENCE

Cached WRITE

CP MODIFY

-DEST CP 0

-DEST CP 1

TB PROBE<00>

This bit specifies a TB PROBE request.

MBOX
MBOX CYCLE CONDITION CODE MICROWORD

ADDRESS FROM IVA<00>

This bit specifies that this is an ADR FROM IBOX VA cycle.

CP LOCK<00>

This bit specifies that this is a CP LOCK request.

ACCESS VIOLATION CHECK<00>

This bit enables the access violation TRAP when a READ CHECK or WRITE CHECK is required.

READ CHECK<00>

This bit specifies that a read access violation check is required, and is asserted with ACCESS VIOLATION CHECK.

EN PAGE BOUNDARY CROSS CHECK<00>

This bit enables the Crossing Page Boundary Trap.

ENABLE ALIGN CHECK<00>

This bit enables longword alignment references to cause a Longword Alignment trap.

CACHE READ<00>

This bit is asserted whenever the Mbox or IBox requests a read from the cache.

-IO ACC ENABLE<00>

When this bit is reset (asserted low) it will enable a trap on IBUS or CP port read requests to IO space.

DEST CP<1:0>

This bit specifies the destination of the MD RESPONSE in the current request. The field has the following encoding:

0 = IBE
1 = CP FETCH
2 = ESOS
3 = TOY FROM CP

CP MODIFY<00>

This bit specifies a CP MODIFY request.

MBOX CHECK<00>

This bit specifies that a write access violation check is required. The bit is asserted with ACCESS VIOLATION CHECK. It is also used whenever an X bit check is required, and is asserted with CACHE WRITE.

CACHE WRITE<00>

This bit is asserted whenever the Ebox or Ibox requests a write to the cache.

-MD WRITE<00>

When reset (asserted low) this bit inhibits Xbox alignment checks for EBox requests on the first reference.

Figure 14-15 XBus Register Read/Write Data Paths

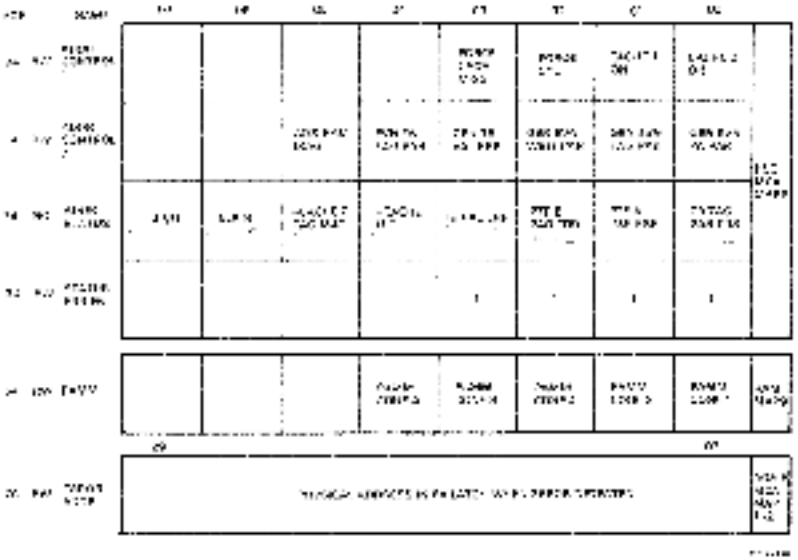


Figure 14-16 NAP Module MBox Registers

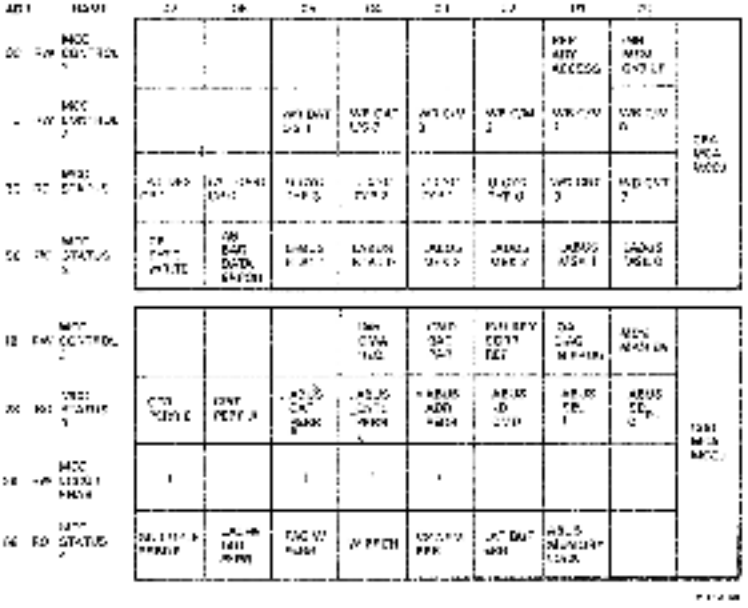


Figure 14-17 PCU Module MBox Registers

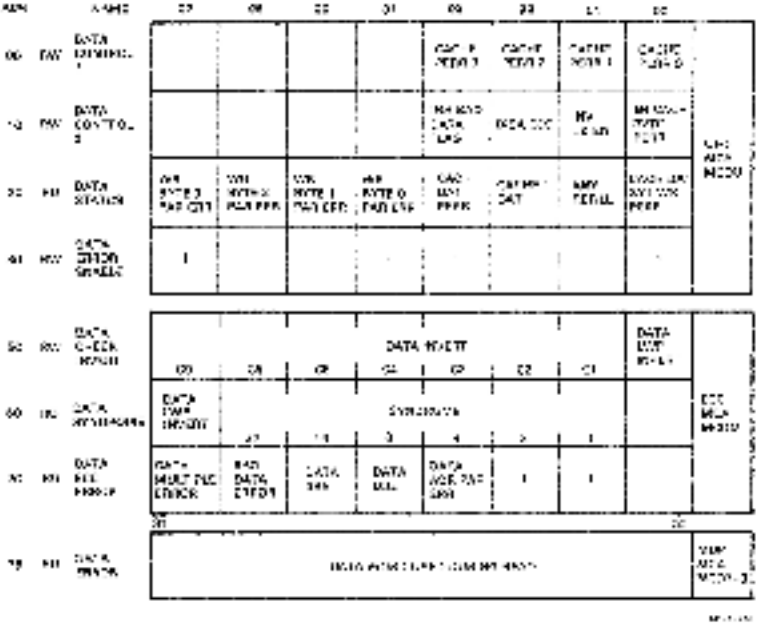


Figure 14-18 NCD Module MBox Registers

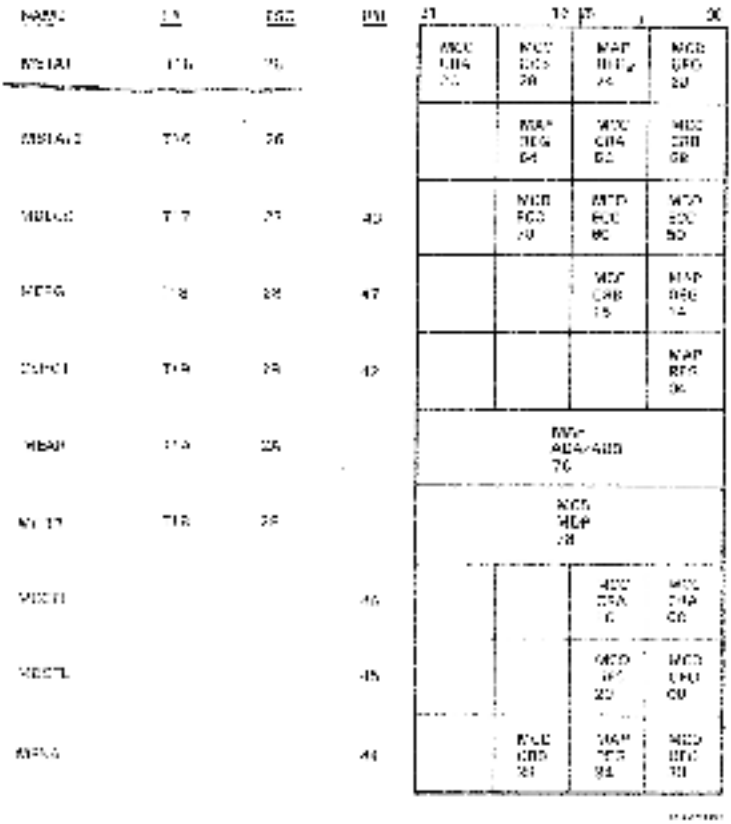


Figure 14-19 MBox Microcode Assembly of MBox Registers

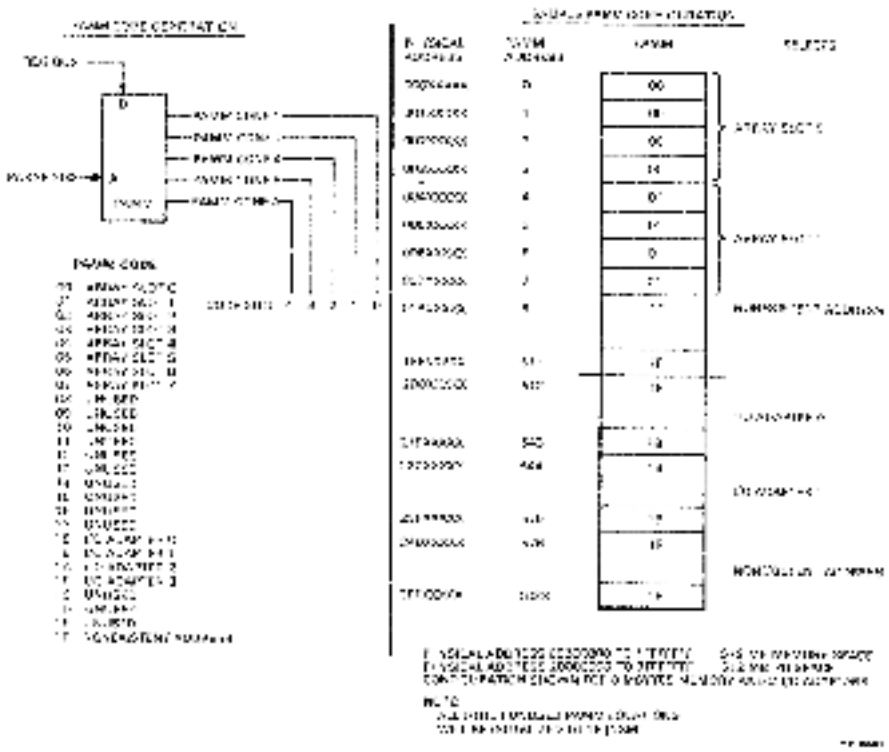


Figure 14-18 Physical Address Memory Map

Figure 14-21 ABus Interface Block Diagram

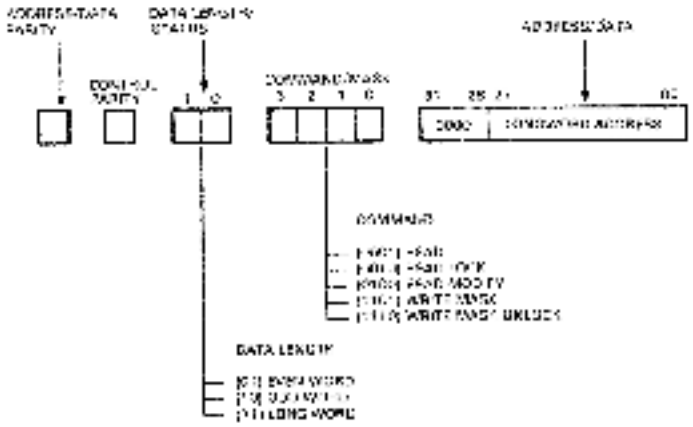


Figure 14-27 ABus CPU Command/Address Cycle Format

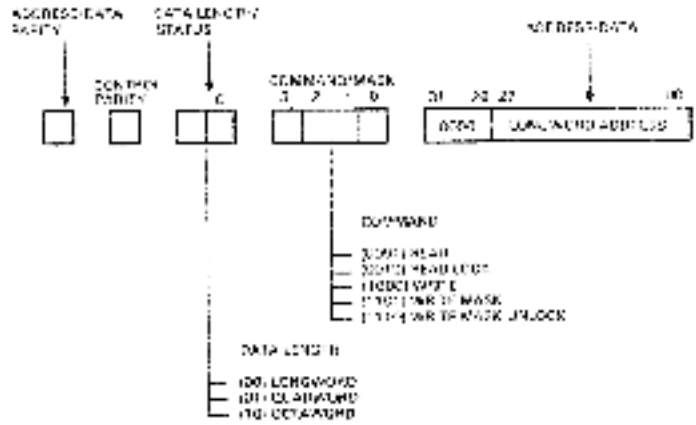


Figure 14-27 ABus DMA Command/Address Cycle Format

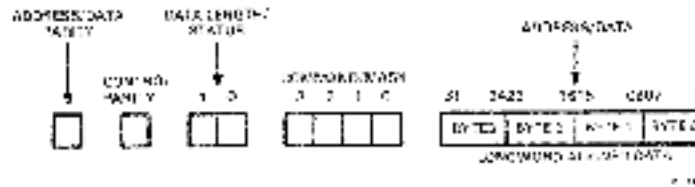


Figure 14-28 ABus Read/Write Cycle Format

Table 14-3 ABUS Signal Pin Location

Signal Name	SIM J	SBA A/B MJ 1	J Conn	J Conn	MCD 16	MAP 17	MCT 18 MJ 2
		ABUS Backplane		CPU Backplane			
ABUS ADR DATA PTY H	C32	C30	J1603	J1604	C30	B31	
ABUS CMD MSK 0 H	B15	C42	J1621	J1622			C10
ABUS CMD MSK 1 H	B16	C47	J1625	J1626			C12
ABUS CMD MSK 2 H	B17	C40	J1615	J1616			C06
ABUS CMD MSK 3 H	B20	C51	J1631	J1632			C14
ABUS CPU BUF ZONE E	C30	C51	J1641	J1642			B36
ABUS CPU BUF BADOR H	C42	C46	J1623	J1624			C09
ABUS CTRL PTY H	C33	C43	J1715	J1716			C37
ABUS DATA ADDRS 0 H	B32	C07	J1419	J1434	A05	A05	
ABUS DATA ADDRS 1 H	B29	C09	J1435	J1436	A07	A07	
ABUS DATA ADDRS 2 H	C28	C09	J1437	J1510	A09	B09	
ABUS DATA ADDRS 3 H	C30	C10	J1439	J1440	A12	A11	
ABUS DATA ADDRS 4 H	C32	C11	J1441	J1442	A20	A19	
ABUS DATA ADDRS 5 H	C34	C14	J1443	J1444	A21	A21	
ABUS DATA ADDRS 6 H	B24	C10	J1511	J1512	A47	A47	
ABUS DATA ADDRS 7 H	B26	C40	J1513	J1514	A60	A59	
ABUS DATA ADDRS 8 H	B25	C21	J1515	J1516	A64	A63	
ABUS DATA ADDRS 9 H	B28	C22	J1517	J1518	A65	A65	
ABUS DATA ADDRS 10 H	B27	C25	J1519	J1520	A70	A69	
ABUS DATA ADDRS 11 H	B30	C26	J1521	J1522	B09	B09	
ABUS DATA ADDRS 12 H	B19	C27	J1523	J1524	B15	B15	
ABUS DATA ADDRS 13 H	B22	C28	J1525	J1526	B17	B17	
ABUS DATA ADDRS 14 H	B21	C29	J1527	J1528	B19	B19	
ABUS DATA ADDRS 15 H	B31	C30	J1529	J1530	B67	B67	
ABUS DATA ADDRS 16 H	B34	C32	J1543	J1444	B57	B57	
ABUS DATA ADDRS 17 H	B35	C35	J1601	J1602	B59	B59	
ABUS DATA ADDRS 18 H	B36	C36	J1603	J1604	B61	B61	
ABUS DATA ADDRS 19 H	B37	C37	J1605	J1606	C71	C37	
ABUS DATA ADDRS 20 H	B39	C39	J1607	J1608	B73	C31	
ABUS DATA ADDRS 21 H	B39	C39	J1613	J1614	C08	C07	
ABUS DATA ADDRS 22 H	B40	C41	J1619	J1620	C10	C09	
ABUS DATA ADDRS 23 H	B41	C49	J1627	J1628	C11	C91	
ABUS DATA ADDRS 24 H	B47	C53	J1629	J1640	C21	C01	
ABUS DATA ADDRS 25 H	B46	C54	J1641	J1642	C24	C77	
ABUS DATA ADDRS 26 H	B45	C55	J1721	J1722	C54	C53	
ABUS DATA ADDRS 27 H	B40	C57	J1723	J1723	C56	C55	
ABUS DATA ADDRS 28 H	B50	C64	J1725	J1726	C57	B20	
ABUS DATA ADDRS 29 H	B52	C71	J1731	J1732	C63	B61	
ABUS DATA ADDRS 30 H	C42	C73	J1733	J1734	C65	C66	
ABUS DATA ADDRS 31 H	B41	C74	J1735	J1736	C69	A77	
ABUS OCADD L		AB33	J1509	J1510			A0252
ABUS DBADJ L		A033	J1501	J1502			A0240
ABUS DMA DONE 0 H		C0364	J1717	J1718			C28
ABUS DMA DONE 1 H		C0364	J1705	J1706			C29
ABUS DMA DONE 2 H			J1703	J1704			C27
ABUS DMA DONE 3 H			J1707	J1708			C30

Table 14-3 ABus Signal Pin Location (Cont.)

ABUS IPR RETURN 0 H	C18	C26	J1739	J1740	C0535
ABUS IPR RETURN 1 P	C86	C75	J1737	J1738	C0570
ABUS IPR RETURN 2 R	C37	C47	J1741	J1742	C0576
ABUS IPR RETURN 3 T	C36	C78	J1767	J1744	C0578
ABUS IPR RETURN 4 F	C31	C40	J1803	J1501	C0583
ABUS IPR SELECT 0 A		C0190	J1807	J1804	C0594
ABUS IPR SELECT 1 T		C0590	J180	J1802	C0591
ABUS IPR SELECT 2 S			J1805	J1849	C0560
ABUS IPR SELECT 3 D			J1541	J1542	C0562
ABUS LOCK SIGL D B	C40	C40	J1729	J1730	C62
ABUS LBEN START 1 H	C84	C69	J1727	J1728	C60
ABUS MEMORY LOCK E	C09	C05	J1479	J1480	S85
ABUS ASKED CMD F	A48	C04	J1427	J1428	C02
ABUS XR CMD 4	A19	C06	J1431	J1432	C04
MCC ABUS CPU BUS SEL B	C85	C55	J1542	J1604	C20
MCC ABUS DMA ERROR D	B49	C60	J1711	J1712	C10
MCC ABUS IOA SELECT 0 C		C0357	J1701	J1702	C35
MCC ABUS IOA SELECT 1 B		C0555	J1609	J1610	C03
MCC ABUS IOA SELECT 2 1			J1617	J1618	C07
MCC ABUS IOA SELECT 3 A			J1611	J1612	C05
MCC ABUS XBUA OUT D	C08	C05	J1719	J1720	C25
MCC ALORS CTRL 0 1	C31	C62	J1713	J1714	C15
MCC ADDRS CTRL 1 L	C29	C59	J1709	J1710	C09
SP ABUS IOA REQUEST 0 E		C0452	J1633	J1634	C20
SP ABUS IOA REQUEST 1		C0552	J1629	J1630	C11
SP ABUS IOA REQUEST 2 B			J1631	J1632	C13
SP ABUS IOA REQUEST 3 D			J1627	J1628	C19

NOTE

1. The pin number under the column titled "SBA 3/5" refers to the signal pin for both slots 3 and 5 unless it is 5 characters, in which case it designates the SBA module in one particular slot.

2. Under the column titled "MCC 16", if the pin designation is 5 characters, then the signal is not connected to the MCC module, but rather to another module in the slot identified by the pin number.

3. The numbers under the module designations refer to the slot number(s) for the particular modules, as slot 1 for the STA module (ABus backplane) and slot 16 for the MCC module (CPU backplane).

14.7 ABUS INTERFACE SIGNALS

The following sections provide the ABus interface signals and their meanings.

- DATA/ADDRESS <31:00>—During Command/Address cycles, these bi-directional bits will carry a 28-bit physical address between the KBox and SBIA. The most significant 4 bits will be 0.

  During read or write data cycles, these bi-directional bits carry the data between the KBox and SBIA.

- DATA/ADDRESS PARITY—Odd parity computed over the 32 bit data/address field.

- COMMAND/MASK <03:00>—During a Command/Address cycle, these bi-directional four bits indicate the ABus command to the KBox or SBIA according to Table 14-4.

Table 14-4 ABus Commands

Command/Mask <03:00>	Command	Command Application
0001	Read	CPU/DMA
0010	Read lock	CPU/DMA
0100	Read modify	CPU
1000	Write	DMA
1101	Write mask	CPU/DMA
1110	Write mask unlock	CPU/DMA

During write data cycles these bits are the MASK bits, and are used to specify which byte(s) are written. If a bit is set, the corresponding byte is written.

During read data return cycles, the MASK bits are not used.

- DATA LENGTH/STATUS <01:00>—These bits indicate the data length during Command/Address cycles according to Table 14-5 and Table 14-6.

Table 14-5 Length/Status for CPU Command/Address Cycle

Length/Status <01:00>	CPU Command/Address Function
00	Read Word
10	Odd Word
11	Longword

Table 14-6 Length/Status for DMA Command/Address Cycle

Length/Status <01:00>	DMA Command/Address Function
00	Longword (4 bytes)
01	Quadword (8 bytes)
10	Octaword (16 bytes)

NOTE

The SBIA only supports Longword and Quadword transfers.

During Data cycles, these bits indicate the status of the data. The STATUS bits only have meaning during a CPU read data cycle. Normally zero, they will be "1." to indicate bad data, if the SBIA received Read Data Subscribe (RDS). Or on the SBI read data cycle (SBI data cycle mask bits = 0010).

- CONTROL PARITY--Odd parity computed over the Length/Status and Command/Mask bits.

- MSKED CMD--This bit, when set, tells the MBox that the ABus command is for a masked operation. It allows the MBox microcode to branch before decoding the command field in the command/address.

- ABUS WR CMD--Like the MSKED CMD, ABUS WR CMD allows the MBox microcode to branch before decoding the command field in the command/address. This bit indicates that the ABus command is for a write operation.

- CLK SBA[N] CLOCKS 161 B[D]--Each I/O adapter receives clock signals from the SBA6 system clock module to control the SBUS interface logic.

- CLK SBA[N] RESET--A reset signal to each of the I/O adapters used to initialize the clock generation and distribution logic.

- IOA REQUEST <03:00>--One line from each I/O adapter used to request the MBox to service a DMA.

- DMA DONE <03:00>--A signal to each of the I/O adapters to signify that the MBox has completed the DMA transfer. Only the I/O adapter that had a DMA being serviced will receive the DMA DONE. In case of a DMA error, DMA DONE will be sent with DMA ERROR.

- DMA ERROR--A signal to each of the I/O adapters to indicate that the MBox has detected an error while processing the DMA. DMA ERROR is sent at the same time as DMA DONE.

- ABUS ADDRS CTRL <01:00>—These two signals are used in the selected I/O adapter to control the loading, holding, or incrementing of the LC022 register file address, according to Table 14-7. These bits are asserted low, and in the table, a logic 1 indicates a deasserted signal.

Table 14-7 Address Control of the LC022 Register File

ADDRS CTRL <01:00>		LC022 Function
0	0	Hold address
0	1	Increment address
1	0	Not used
1	1	Load address

- XBOX OUT—XBox OUT controls whether the I/O adapter register file, for the I/O adapter selected by ABUS DEVICE, will be read or written. It is also used with ABUS CPU BUF SEL to control the two least significant bits of the LC022 address being loaded by the XBox (see Table 14-8). If ABUS XBOX OUT is asserted data will be driven by the XBox to the I/O adapter. If it is not asserted, the I/O adapter will be sending data to the XBox.

- ABUS CPU BUF SEL—This signal is used along with ABUS XBOX OUT, to control the register file address loaded by the XBox in the selected I/O adapter (see Table 14-8).

Table 14-8 LC022 Register File Address Control

ABUS CPU BUF SEL	ABUS XBOX OUT	Register File LC0 Address	Contents
0	0	**00	Prepare to read the DMA C/A
0	1	**01	Prepare to write a CPU C/A or an XBox diagnostic cycle
1	0	0011	Prepare to read the CPU read data return word
1	1	0010	Prepare to write the CPU C/A

** The upper two bits are determined by the DMA transaction to the first requested service.

- CPU BUS BUSY--Asserted by the I/O adapter to inform the MBox that a CPU read or write transaction is complete. A CPU write is complete when the MBox has acknowledged the command/address and write data. A read is complete when the read data is placed in the C.BUF register file. CPU BUS BUSY will be asserted if CPU BUS ERROR is asserted.

- CPU BUS ERROR--CPU BUS ERROR is asserted by the selected I/O adapter to indicate to the MBox that an error was detected during a CPU initiated I/O transaction. It indicates that the I/O adapter aborted the command.

- MEMORY LOCK--A bi-directional line between the MBox and I/O adapters to signal that an interlock operation is in progress.

- ABUS IPR RETURN <04:00>--The encoded value of the highest priority interrupt the I/O adapter has pending, when the I/O adapter is polled by the MBox with the assertion of ABUS IPR SELECT[X].

- ABUS IPR SELECT[X]--ABUS IPR SELECT is used by the MBox to poll each of the I/O adapters for pending interrupts. The MBox polls the adapters in modulo 4 order.

- ABUS ENABLE--This bit is controlled by the console with PUSH(<01>). When this bit is asserted, the I/O adapters are enabled.

- ABUS DEAD[X]--This bit is asserted as a result of the assertion of BAT FAIL and used to forward a reboot request from another processor or the CI to the console, or to inform the console of an SST power failure.

- AC LO--A signal from the ENV to the I/O adapters used to indicate impending power failures. It is used by the I/O adapter to assert BUS ERR (<4>) on the SBI.

- DC LO--DC LO from the ENV is used to assert BUS SbI DatD and force an initialization within the I/O adapters.

14.8 MBOX DIAGNOSTICS

14.8.1 MBox MicroPC and ABus Control Testpoints

Table 14-9 is a list of MBox MicroPC and ABus control testpoints that may be used when troubleshooting MBox microcode and ABus control problems.

Table 14-9 MBox MicroPC and ABus Control Testpoints

Channel	Signal Name	Print	Section/Slot/Pin
F	MUPC 0 E (LSB)	MCC7-9	Bxx.17 (MCL)
E	MUPC 1 E	MCL2-8	Bxx.14 (ECL)
D	MUPC 2 E	MCC7-9	Bxx.15 (MCL)
C	MUPC 3 E	MCC5-1	Bxx.16 (ECL)
B	MUPC 4 E	MCC3-M	Bxx.17 (MCL)
A	MUPC 5 E	MCL4-8	Bxx.09 (ECL)
9	MUPC 6 E	MCC3-3	Bxx.10 (ECL)
8	MUPC 7 E (MSB)	MCC0-8	Bxx.11 (ECL)
7	ABUS IOA REQ 0 H	83A5	C03.53 (MCL)
6	ABUS IOA SEL 0 H	83A5	C05.52 (ECL)
5	ABUS IOA SEL 1 H	83A5	C03.5A (MCL)
4	ABUS ADR CTL 1 H	63A5	C05.56 (ECL)
3	ABUS ADR CTL 0 H	83A5	C03.59 (MCL)
2	ABUS MBOX DAT H	83A5	C03.62 (ECL)
1	ABUS MB CMD H	83A5	C03.66 (ECL)
0	----	-----	------
CLK	SBIA FAULT OR ERROR H	8831	B02.49 (TTL)
	CLOCK	MCCK	B31.12 (ECL)

NOTE

1. "Bxx" indicates any RAM chip for MBox microcode.

2. To display the MBox uPC in HEX, MUPC 7 is the MSB and MUPC 0 is the LSB.

14.9.2 ABus/SBIA Testpoints

Table 14-10 contains a list of ABus signals and SBI FAULT, which can be used when troubleshooting SBI FAULT/ABus problems.

Table 14-10 ABus/SBIA Testpoints

Channel	Signal Name	Print	Section/Slot/Pin
0	SA31 FAULT OR ERROR H	88A	B02.49 (TTL)
1	ABUS IOA REQ H	83A	C03.52 (ECL)
2	ABUS IOA SELECT H	83A	C03.56 (ECL)
3	ABUS ADR CTL 1 L	83A	C03.59 (ECL)
4	ABUS ADR CTL 0 L	83A	C03.62 (ECL)
5	ABUS MBOX DAT H	83A	C03.65 (ECL)
6	ABUS MB CMD H	83A	C03.06 (ECL)
7	ABUS PARITY ERR H	83A	C03.04 (ECL)
8	ABUS DATA/ADR 00 H	83A	C03.07 (ECL)
9	ABUS DATA/ADR 01 H	83A	C03.09 (ECL)
A	ABUS DATA/ADR 02 H	83A	C03.0C (ECL)
B	ABUS DATA/ADR 03 H	83A	C03.13 (ECL)
C			
D			
E	ABUS BUS ERROR H	73A	C03.46 (ECL)
F			

14.8.3 CP Port Testpoints

Table 14-11 contains a list of CP port testpoints.

Table 14-11 CP Port Testpoints

Channel	Signal Name	Print	Section/Slot/Pin
0	MCC DP PA ACK A H	MCC6	B18.39
1	MCC EBOX PA ACK A H	MCC6	B18.41
2	MCC IOP PA ACK H	MCC6	B10.37
3	MCC KD BSSP C H	MCC5	C18.75
4	MCC PORT STAT CODE 3 H	MCC7	B18.20
5	MCC PORT STAT CODE 2 H	MCC7	B10.35
6	MCC PORT STAT CODE 1 H	MCC7	B18.11
7	MCC PORT STAT CODE 0 H	MCC7	B18.18

CHAPTER 15

SBIA

15.1 SBIA GENERAL INFORMATION

The SBIA is configured/cleared during CPU initialization by the INIT/PARM console command. This command is required to reset the SBIA. The SBIA must be configured prior to attempting any access to the SBI for an UNJAM or depositing/examining nexus control and status registers. Therefore, to reset the I/O world, the following sequence of console commands must be executed.

- >>>INIT/PARM
- >>>UNJAM
- >>>INIT

For an example of SBI SILO interpretation, refer to Chapter 20, SBIA SBI SILO Register description.

15.2 SBIA JUMPER SETTINGS

The SBIA has two DC101 chips used for SBI arbitration. One DC101 is used to gain control of the SBI for CPU deposits/examines of SBI nexus control and status registers. The other DC101 is used to gain control of the SBI to transfer the DMA read word to the appropriate nexus. Table 15-1 shows the jumper information for the SBIA DC101s.

Table 15-1 SBIA DC101 TR Jumpers

TR SBL 8 4 2 1	DEVICE TR	Transmit TR Jumper Needed	Jumper from XMIT TR (, DC1* to BUS SBI TR __ on Pin__	
L L L L	16	NO	--	---
L L L H	15	Yes	15	C81
L L H L	14	Yes	14	C77
L L H H	13	Yes	13	C73
L H L L	12	Yes	12	C75
L H L H	11	Yes	11	C71
L H H L	10	Yes	10	C69
L H H H	09	Yes	09	C67
H L L L	08	Yes	08	C65
H L L H	07	Yes	07	C63
H L H L	06	Yes	06	C59

Table 15-1 SE1A DC101 TR Jumpers (Cont)

IR SEL 0 1 2 3	DEVICE TR	Transmit IR Jumper Needed	Jumper from XMIT TR J.. C83* to SE2 SBI TR __ on Pin___	
H L H H	01	Yes	05	C57
H H L L	04	Yes	04	C55
H L L H	03	Yes	03	C52
H H H L	02‡	Yes	02	C53
H E E E	01†	Yes	01	C47

*Jumper connections made on SBI backplane slots
†Normal configuration for DPL DC101
‡Normal configuration for DMA DC101

15.1 SE1A BLOCK DIAGRAM

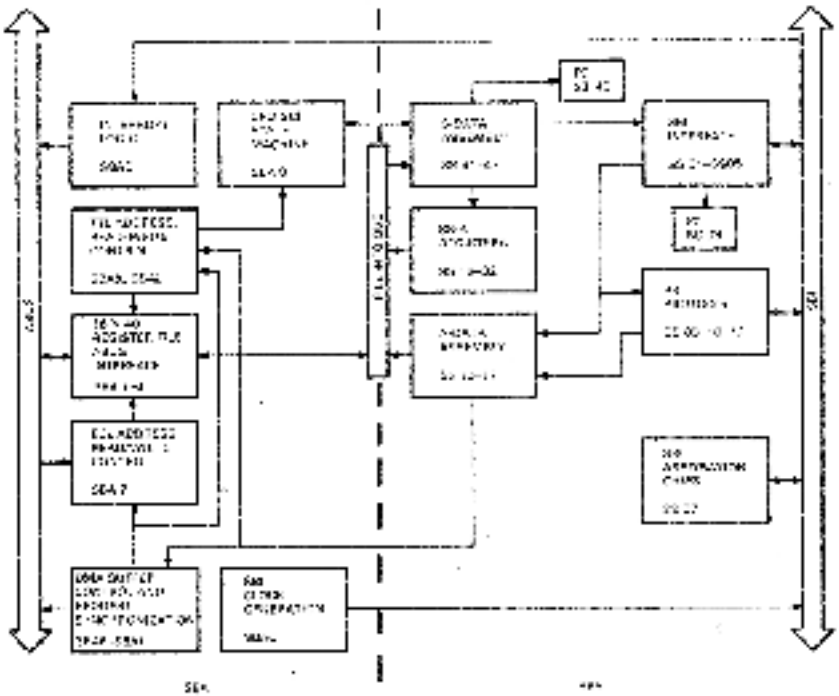


Figure 15-1 SE1A Block Diagram

15.4 VAX 86XX PHYSICAL MEMORY ADDRESS ALLOCATION

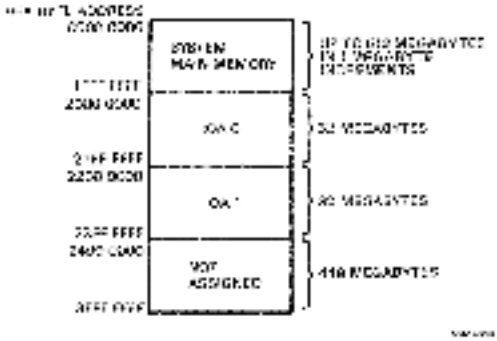


Figure 15-2 VAX 86XX Physical Memory Address Allocation

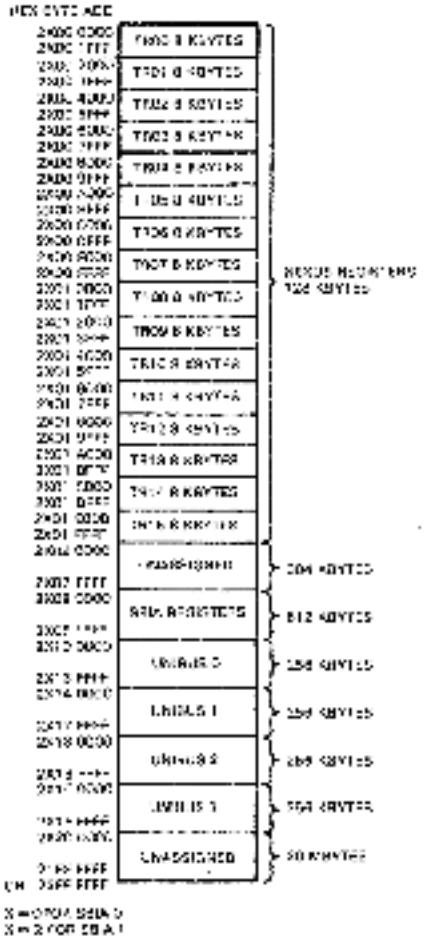


Figure 15-1 I/O Adapter Physical Address Allocation

15.4.1 SBIA Register Addresses

Table 15-2 SBIA Register Addresses

Register Name	Hex Byte Address
Configuration Register	2X080000
Control Status Register	2X080004
Error Summary Register	2X080008
Diagnostic Control Register	2X08000C
DMAI Command/Address Register	2X080010
DMAI ID Register	2X080014
DMAA Command/Address Register	2X080018
DMAA ID Register	2X08001C
DMAB Command/Address Register	2X080020
DMAB ID Register	2X080024
DMAC Command/Address Register	2X080028
DMAC ID Register	2X08002C
SBI Silo	2X080030
SBI Error Register	2X080034
SBI Timeout Address Register	2X080038
SBI Fault/Status Register	2X08003C
SBI Silo Comparator	2X080040
SBI Maintenance Register	2X080044
SBI Regen Register	2X080048
SBI Quadclear Register	2X08004C
Vector Registers	2X080080
↓	↓
Vector Registers	2X08013C

NOTE

For SBIA 0, X = 0 and for SBIA 1, X = 2

15.5 SBI PROTOCOL

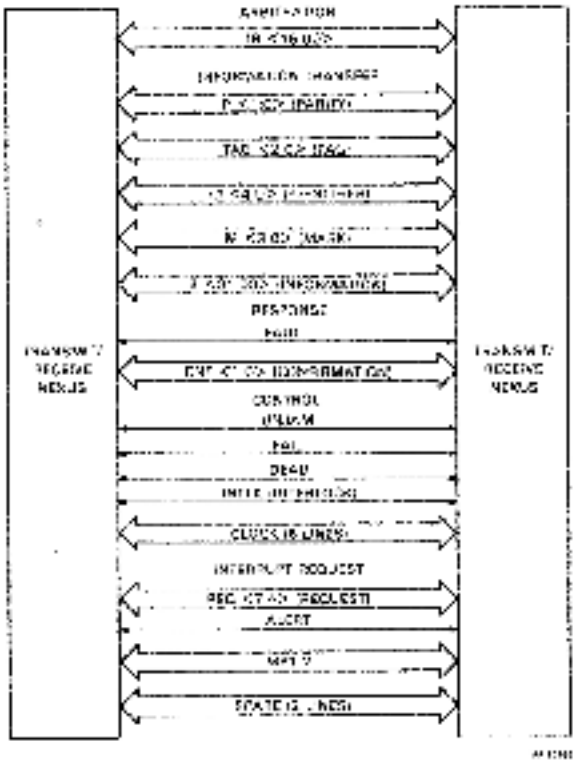


Figure 15-4 SBI Signal Names

Table 15-3 SBI Signal Names and Description

Arbitration Group

Arbitration Field (ARB<15:00>) -- Establishes a fixed priority among nexus for access to and control of the information transfer path.

Information Transfer Group

Information Field (B<31:00>) -- Bidirectional lines that transfer data, command/address, and interrupt information between nexus.

Mask Field (M<03:00>) -- Encoded to indicate that a particular byte within the data field is to be read or written. With the read data, the mask bits indicate if the data is correct or in error.

Identifier Field (ID<02:00>) -- Indicates the logical source or destination of information contained in B<31:00>.

Tag Field (TAG<02:00>) -- Determines if the SBI cycle is for a command/address, read data, write data, or IDX.

Response Group

Confirmation Field (CNF<01:00>) -- The receiving nexus specifies its response to an SBI cycle:

1. 00 = no response
2. 01 = acknowledge
3. 10 = busy
4. 11 = error

Function Field (F<03:00>) -- Specifies the command code. The function field is bits <31:28> of the command/address (Figure 15-6).

Parity Field (P<01:00>) -- Indicates even parity over the SBI information. P0 is computed over the information in B<31:00> while P1 is computed over M<03:00>, TR<04:00>, and PAR<02:00> (Figure 15-5).

Interrupt Request Group

Request Field (RQ<07:04>) -- Each signal represents a level of priority whereby a nexus requests interrupt service.

Alert Field (ALERT) -- A signal that allows SBI memory to interrupt due to a power loss.

Table 15-3 SBI Signal Names And Description (Cont)

Control Group

Clock Field [CLOCK] -- Six control lines that provide the clock signals necessary to synchronize SBI activity.

Fail Field [FAIL] -- The assertion of AC LO within a nexus causes the assertion of SBI FAIL.

Dead Field [DEAD] -- The assertion of DC LO within a nexus causes the assertion of SBI DEAD.

Unjam Field [UNJAM] -- A reset signal to all SBI nexus.

Interlock Field [INTLK] -- A signal that indicates that a shared location is being modified by a nexus.

Unused Signals

Multiprocessor [MP<02,01>] -- Two unused lines

Spare [SPARE<01:00> -- Two additional unused lines

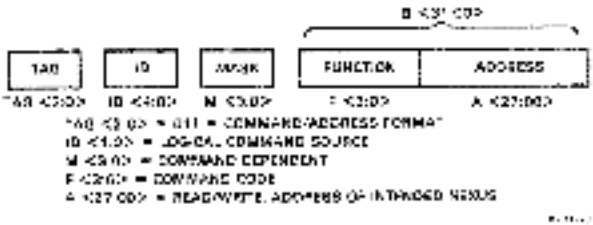
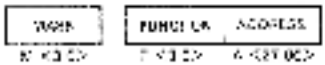


Figure 15-5 SBI Command/Address Format



MASK USE	FUNCTION CODE	FUNCTION USE/VALUE
IGNORED	0000	RESERVED
USED	0001	READ MASKED
USED	0010	WRITE MASKED
IGNORED	0011	RESERVED
USED	0100	INTERLOCK READ MASKED
IGNORED	0101	RESERVED
IGNORED	0110	RESERVED
USED	0111	INTERLOCK WRITE MASKED
IGNORED	1000	EXTENDED READ
IGNORED	1001	RESERVED
IGNORED	1010	RESERVED
USED	1011	EXTENDED WRITE MASKED
IGNORED	1100	RESERVED
IGNORED	1101	RESERVED
IGNORED	1110	RESERVED
IGNORED	1111	RESERVED

Figure 15-6 SBI Command Codes

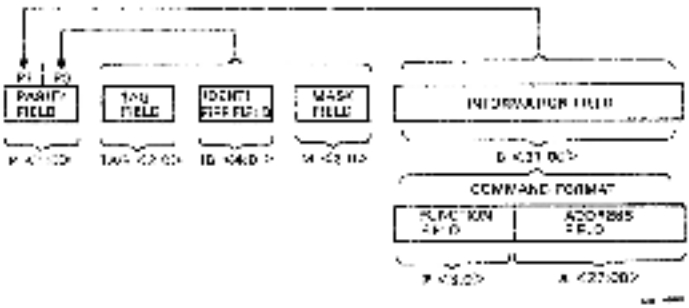


Figure 15-7 SBI Parity Field Configuration

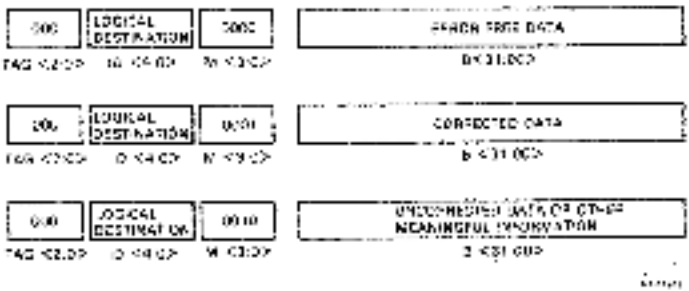


Figure 15-8 SBI Read Data Formats



Figure 15-9 SBI Write Data Format

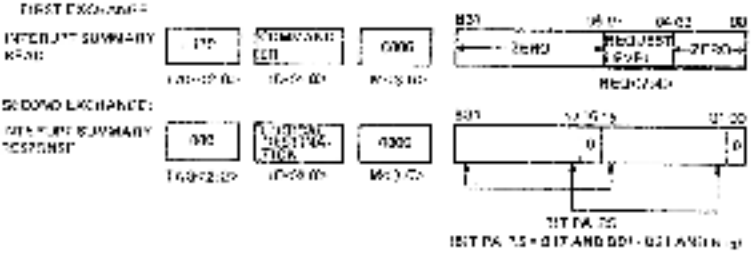


Figure 15-10 SBI Interrupt Summary Formats

15.6 SBI -- I/O BACKPLANE INTERCONNECTIONS

Table 15-4 lists the cable connectors associated with the SBI cables from the ABus backplane to the I/O backplane and for expansion of the SBI from the I/O backplane. See also Figure 15-11 for locations of the cable connectors. Table 15-5 lists the SBI signals along with their backplane pin for the cable from the ABus backplane to the I/O backplane and from the I/O backplane to an SBI expansion cabinet.

Table 15-4 SBI Cable Interconnections

	(SBI Expansion) I/O Backplane		ABus Backplane to I/O Backplane I/O Backplane		ABus Backplane	
Row	SBI1 OUT	SBI0 OUT	SBI1 IN	SBI0 IN	SBI0	SBI1
A	J47	J41	J19	J25	J25	J19
B	J48	J42	J20	J26	J26	J20
C	J49	J43	J21	J27	J27	J21
D	J50	J44	J22	J28	J28	J22
E	J51	J45	J23	J29	J29	J23
F	J52	J46	J24	J30	J30	J24

Figure 15-11 I/O Backplane Assemblies

Table 15-5 SBI Signal and Backplane Pins

Signal Name Note 1	ABus Out Note 2	I/O In Note 2	I/O Out Note 3	Nexus Note 4
BD0	J25 B	J25 A	J41 B	AB1
BD1	J25 D	J25 C	J41 D	AC1
BD2	J25 J	J25 H	J41 J	AD1
BD3	J25 R	J25 P	J41 R	AE1
BD4	J25 BB	J25 AA	J41 BB	AM1
BD5	J25 FF	J25 EE	J41 FF	AN1
BD6	J25 NN	J25 MM	J41 NN	AP1
BD7	J25 DD	J25 CC	J42 DD	AP2
BD8	J25 LL	J25 KK	J41 LL	AR2
BD9	J25 RR	J25 PP	J41 RR	AT2
BD10	J25 TT	J25 SS	J41 TT	AU1
BD11	J25 VV	J25 UU	J41 VV	AV2
BD12	J26 L	J26 K	J42 L	BF2
BD13	J26 T	J26 S	J42 T	BH2
BD14	J26 N	J26 M	J42 N	BJ1
BD15	J26 X	J26 W	J42 X	BC2
BD16	J26 JJ	J26 HH	J41 JJ	BK1
BD17	J26 NN	J26 K	J42 L	BM2
BD18	J26 TT	J26 SS	J42 TT	BU1
BD19	J26 VV	J26 UU	J42 VV	BV2
BD20	J27 D	J27 C	J43 D	CD1
BD21	J27 J	J27 H	J43 J	CC1
BD22	J27 N	J27 M	J43 N	CH1
BD23	J27 P	J27 R	J43 P	CD2
BD24	J27 BB	J27 AA	J43 BB	CM1
BD25	J27 FF	J27 EE	J43 FF	CN1
BD26	J27 LL	J27 KK	J43 LL	CP1
BD27	J27 DD	J27 CC	J43 DD	CP2
BD28	J27 JJ	J27 HH	J42 JJ	CR2
BD29	J27 NN	J27 MM	J43 NN	CT2
BD30	J27 RR	J27 PP	J43 RR	CU1
BD31	J27 VV	J27 UU	J43 VV	CV2
FAIL	J27 TT	J27 SS	J44 TT	CV1
BEAD	J28 D	J28 C	J44 D	DA1
ID0	J28 DD	J28 CC	J44 DD	DD2
ID1	J28 LL	J28 KK	J44 LL	DG2
ID2	J28 RR	J28 PP	J44 RR	DJ2
ID3	J28 TT	J28 SS	J44 TT	DJ1
ID4	J28 VV	J28 UU	J44 VV	DU2
MD0	J28 B	J28 A	J44 B	DB1
MD1	J28 F	J28 E	J44 F	DC1
MD2	J28 L	J28 K	J44 L	DD1
MD3	J28 Q	J28 P	J44 Q	DD2

Table 15-5 SBI Signal and Backplane Pins (Cont.)

P0	J28 N	J28 M	J44 N	DV2
P1	J28 T	J28 S	J44 T	DA2
SPARE 0	J28 R	J28 P	J44 R	
SPARE 1	J28 V	J28 U	J44 V	
TAC0	J28 BP	J28 AA	J44 W	DW1
TAG1	J28 VV	J28 UU	J44 PP	DX1
TAG2	J28 MM	J28 NN	J44 MM	DF1
ALERT	J29 LL	J29 KK	J45 LL	BD2
CNFU	J29 RR	J29 PP	J45 NN	BS2
CNF1	J29 TT	J29 UU	J45 VV	BT2
PARITY	J29 SS	J29 RR	J45 TT	BU1
MF1	J29 FF	J29 EE	J45 JJ	BW1
MF2	J29 DD	J29 CC	J45 DD	BX1
PCLK H	J29 K	J29 P	J45 R	BS2
PCLK L	J29 L	J29 V	J45 V	BT1
PDCLA H	J29 K	J29 N	J45 K	BV1
PDCLT L	J29 BB	J29 AA	J45 BB	BW2
REQ 4	J29 U	J29 A	J45 B	BB1
REQ 5	J29 P	J29 E	J45 F	BC1
REQ 6	J29 N	J29 M	J45 N	BD1
REQ 7	J29 D	J29 C	J45 D	BE1
TP H	J29 T	J29 S	J45 U	BF1
TP L	J29 U	J29 T	J45 J	BL2
UNJAM	J29 BA	J29 MM	J45 MM	BP1
IR00	J30 A	J30 A	J46 B	PA1
IR01	J30 D	J30 C	J46 C	PB1
IR02	J30 F	J30 E	J46 F	PC1
IR03	J30 J	J30 H	J46 J	PD1
IR04	J30 L	J30 K	J46 L	PE1
IR05	J30 N	J30 M	J46 M	PF2
IR06	J30 S	J30 S	J46 T	PH2
IR07	J30 V	J30 U	J46 V	BP11
IR08	J30 X	J30 W	J46 X	NJ2
IR09	J30 Z	J30 Y	J46 Z	PM1
IR10	J30 BB	J30 CC	J46 DD	PN1
IR11	J30 JJ	J30 HH	J46 JJ	PP2
IR12	J30 RR	J30 NN	J46 MM	PS2
IR13	J30 LL	J30 KK	J46 LL	PT1
IR14	J30 JJ	J30 BB	J46 TT	PU1
IR15	J30 VV	J30 UU	J46 VV	PW1

NOTES

1. Signal names are prefixed by "BUS SBI".

2. J numbers shown are for SBI 0. For SBI 2, subtract 6.

3. J numbers shown are for SBI 0. For SBI 2, add 6.

4. Standard signal pins for slot 1 of any nexus, including I/O backplane slots 6, 11, and 16.

5. All signals are asserted low when true.

6. All signals are TTL except ECLK, PDCLK, and TP, which are ECL.

7. Nexus power and ground pins are:
   +5 V at A2 and V1
   Gnd at C2, B1, H2, and Y1

15.7 SBI FAULT DEFINITIONS

Table 15-6 lists the SBI faults and their definitions.

Table 15-6 SBI Fault Definitions

Parity Error -- An information path error (generated by one or more nexus when the parity calculated over the received data does not agree with the received parity.

Write Sequence Fault -- When a nexus, which received a write masked, extended write masked, or interlock write masked command on the previous cycle, does not receive the write data in the current cycle.

Unexpected Read Data -- When a nexus receives read data, but did not receive a command for extended read, read masked, or interlock read masked on the previous cycle.

Interlock Sequence Fault -- When a nexus receives an interlock write masked command and interlock has not been set by an interlock read masked command.

Multiple Transmitter Fault -- When a transmitting nexus receives an ID different from the transmitted ID.

15.8 SBI CONFIRMATION AND FAULT DECISION FLOW

Figure 15-12 is a flow chart that indicates how SBI nexus check for SBI faults and the decisions involved with sending SBI confirmation, SBI CNF<3:0>.

Figure 15-12 SBI Confirmation and Fault Decision Flow

CHAPTER 16

SBI

16.1 SBI NEXUS ADDRESS INFORMATION

This chapter contains information on SBI nexus addressing, the CI780, DR780, DW780, and KA780. There will be a section covering the module utilization, block diagrams, and jumper selection for each of the devices.

16.1.1 SBI Nexus Addressing

Table 16-1 contains a list of SBI nexus base addresses as a function of the TR number. The address is listed for each SBIA. If the address for Unibus address space is desired, see Figure 16-3.

Table 16-1 SBI Nexus Base Address

	SBIA 0			SBIA 1	
TR#		Byte Address	Longword Address	Byte Address	Longword Address
1		20002000	80008000	22002000	88008000
2		20004000	80010000	22004000	88010000
3	DW0	20006000	80018000	22006000	88018000
4	DW1	20008000	80020000	22008000	88020000
5	DW2	2000A000	80028000	2200A000	88028000
6	DW3	2000C000	80030000	2200C000	88030000
7		2000E000	80038000	2200E000	88038000
8*	RH0	20010000	80040000	22010000	88040000
9*	RH1	20012000	80048000	22012000	88048000
10*	RH2	20014000	80050000	22014000	88050000
11*	RH3	20016000	80058000	22016000	88058000
12		20018000	80060000	22018000	88060000
13		2001A000	80068000	2201A000	88068000
14		2001C000	80070000	2201C000	88070000
15		2001E000	80078000	2201E000	88078000

*RH780

16.1.2 SBI Nexus Address Generation

An SBI nexus physical byte address (the CPU address) may be calculated using Figure 16-1 and the following steps.

1. Fill in bits <28:25> of Figure 16-1 with the I/O adapter number.

2. Fill in bits <24:21> of Figure 16-1 with the TR number of the nexus.

3. Add the register offset (see the register description in Chapter 20 for the register offset) to the bit combination derived from the first two steps.

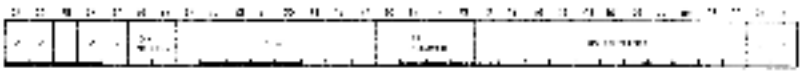


Figure 16-1 SBI Nexus Address Generation

Table 16-2 SBI Nexus Address Generation Bit Descriptions

Bit Number	Description
<26:25>	I/O Adapter Number — 0 = SBIA0 1 = SBIA1 2 = currently unused 3 = currently unused
<16:13>	TR Number — The TR level of the addressed NEXUS
<12:02>	Register Offset — The register offset within the nexus I/O address space. These bits refer to the registers position in SBI address space. When calculating the physical byte address, do not place the register offset into these bits, they are to be added.

Example 16-1 Nexus Address Calculation

Generate the physical byte address of the UBAR IDA Status Register, which has a register offset of 8) for BNO (TR 3) on SBIA 0. Filling in bits <26:25> with 00 (I/O adapter number) and <16:13> with 0011 (TR3) provides a base address of 20006000. Adding the register offset of 8 to the base address gives us a physical byte address of 20006008.

16.1.3 SBI Nexus Interrupt Vector Generation

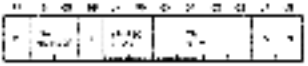


Figure 16-2 SBI Interrupt Vector Generation

Table 16-3 SBI Vector Generation Bit Description

Bit Number	Description
<10:09>	I/O Adapter Number — 0 = SBIA0 1 = SBIA1 2 = currently unused 3 = currently unused
<07:06>	BR(NPR) Level — The interrupt request level or BR level. 00 = BR 4 01 = BR 5 10 = BR 6 11 = BR 7

Table 16-3 SBI Vector Generation Bit Description (Cont)

Bit Number	Description
<05:01>	VR number - the VR number of the NEXUS

Example 16-2 Interrupt Vector generation

An interrupt from a SBI Unibus device on DW0 ITR - 11 will generate an interrupt vector of 10C.

In.2 CI780

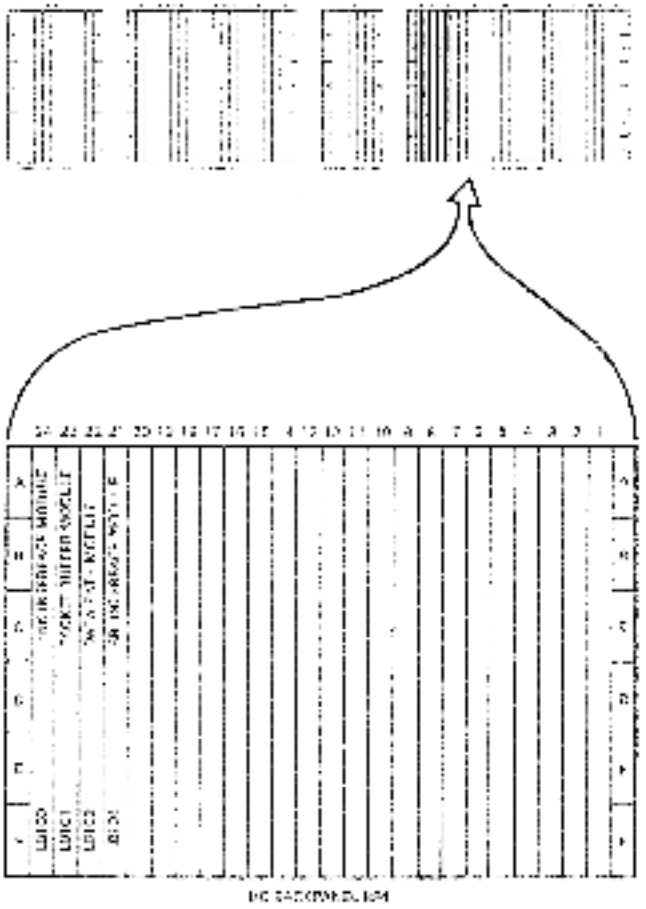


Figure 16-3 CI780 I/O Backplane Module Utilization

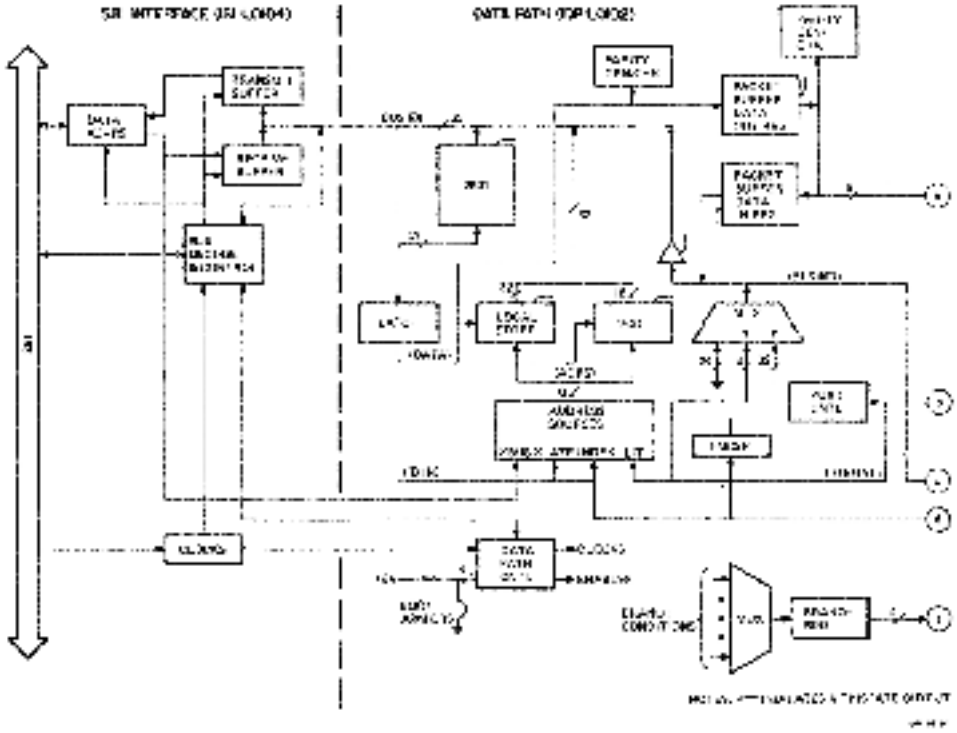


Figure 1e-4 C8T00 Block Diagram (Sheet 1 of 2)

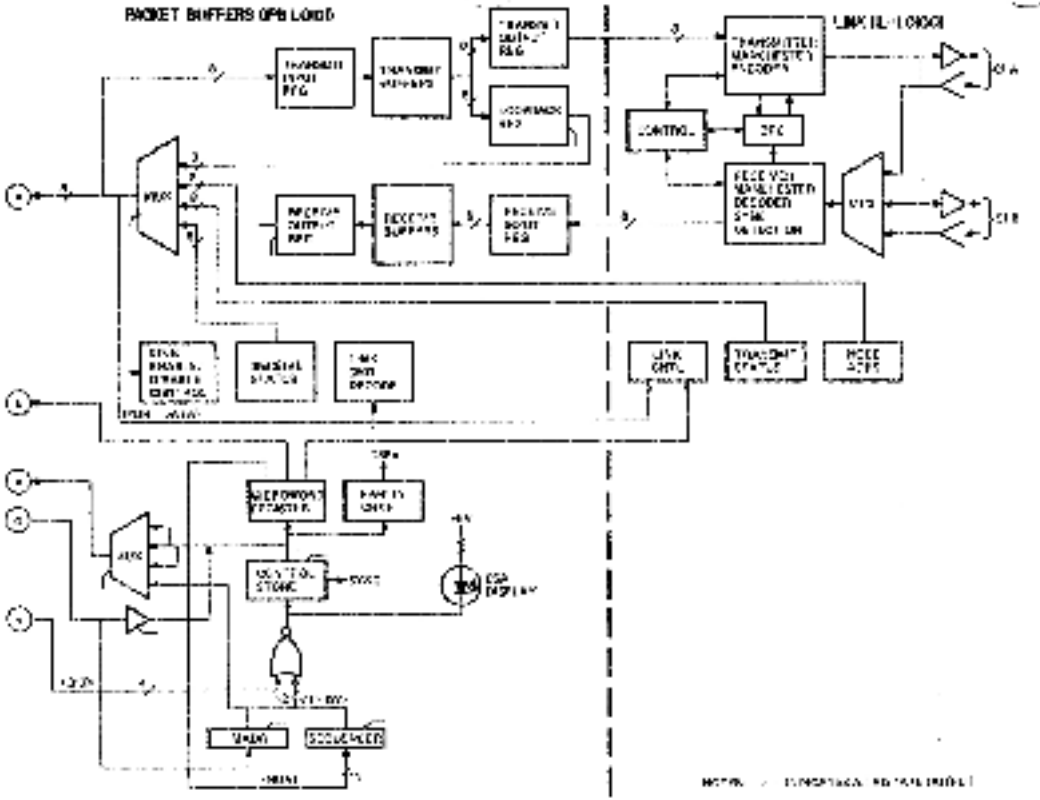


Figure 16-4 CI782 Block Diagram (Sheet 2 of 2)

Figure 16-5 c1760 Backplane Jumper Locations

16.2.1 CI780 Backplane Jumper Settings

Table 16-4 contains a list of the CI780 jumpers with jumper names, and a description of the jumpers.

Table 16-4 CI780 Backplane Jumpers

CI780 Jumpers

J17 or J31	Signal Name	Description
W1	LI Jumper	In = 3010 SBI cycles before CXTMO Out = 512 SBI cycles before CXTMO
W2	TR Jumper D	
W3	TR Jumper C	
W4	TR Jumper A	
W5	PRI Jumper 0	
W6	PRI Jumper 1	
W7	TR Jumper B	
W8	Panic Mode Jumper	In = Panic mode disabled Out = Panic mode enabled
W9	Boot Jumper 5	
W10	Boot Jumper 4	
W11	Reserved	
W12	Boot Jumper 0	
W13	Boot Jumper 2	
W14	Disable Arb	In = No arbitration on CI before transmission Out = Normal CI arbitration
W15	Extend HDR/TRLR	In = Extends header/trailer Out = Normal header/trailer
W16	Alter Delay Time	In = Long delta time Out = Short delta time
W17	Extend ACK Timeout	In = Long timeout Out = Short timeout

NOTE

For the SBI expansion cabinet, use J17, and for the VAX 8600/8650 I/O backplane, use J31.

Interrupt/Priority Level

W5	W6	SBI Level
0	0	4
0	1	5
1	0	6
1	1	7

0 = Jumper out
1 = Jumper in

Table 16-4 CI780 Backplane Jumpers (Cont.)

Boot Timer

W9	W11	W10	W12	Time (Seconds)
0	0	0	0	0000
0	0	0	1	0100
0	0	1	0	0200
0	0	1	1	0300
0	1	0	0	0400
0	1	0	1	0500
0	1	1	0	0600
0	1	1	1	0700
1	0	0	0	0800
1	0	0	1	0900
1	0	1	0	1000
1	0	1	1	1100
1	1	0	0	1200
1	1	0	1	1300
1	1	1	0	1400
1	1	1	1	1500

0 = Jumper in
1 = Jumper out

TR Level

W2	W3	W7	W4	TR Level
0	0	0	0	1
0	0	0	1	2
0	0	1	0	3
0	0	1	1	4
0	1	0	0	5
0	1	0	1	6
0	1	1	0	7
0	1	1	1	8
1	0	0	0	9
1	0	0	1	10
1	0	1	0	11
1	0	1	1	12
1	1	0	0	13
1	1	0	1	14
1	1	1	0	15
1	1	1	1	16

0 = Jumper out
1 = Jumper in

Table 16-4 CI780 Backplane Jumpers (Cont.)

SBI TR Arbitration

For	Wirewrap from CXX-53 (BUS TR L) To
TR1	CX2-57
TR2	CXX-59
TR3	CX2-62
TR4	CXX-65
TR5	CXX-69
TR6	CX2-71
TR7	CXX-73
TR8	CXY-75
TR9	CXX-77
TR10	CX2-81
TR11	CXX-83
TR12	CXX-85
TR13	CXX-86
TR14	CX2-87
TR15	CXX-90

For SBI expansion cab, X = C01
For I/O backplane, X = C24

NOTES

1. The standard configuration is TR4 with all other jumpers out.
2. Other wirewrap needed:
   a. Expansion cabs: Jumper B05-05 to B05-06
   b. I/O backplanes: Jumper B24-05 to B24-06
3. The jumper selection for TR level must match the wirewrap for TR arbitration.
4. TR0 is reserved for holding the SBI.

16.3 DR780



Figure 16-6 DR780 Module Utilization

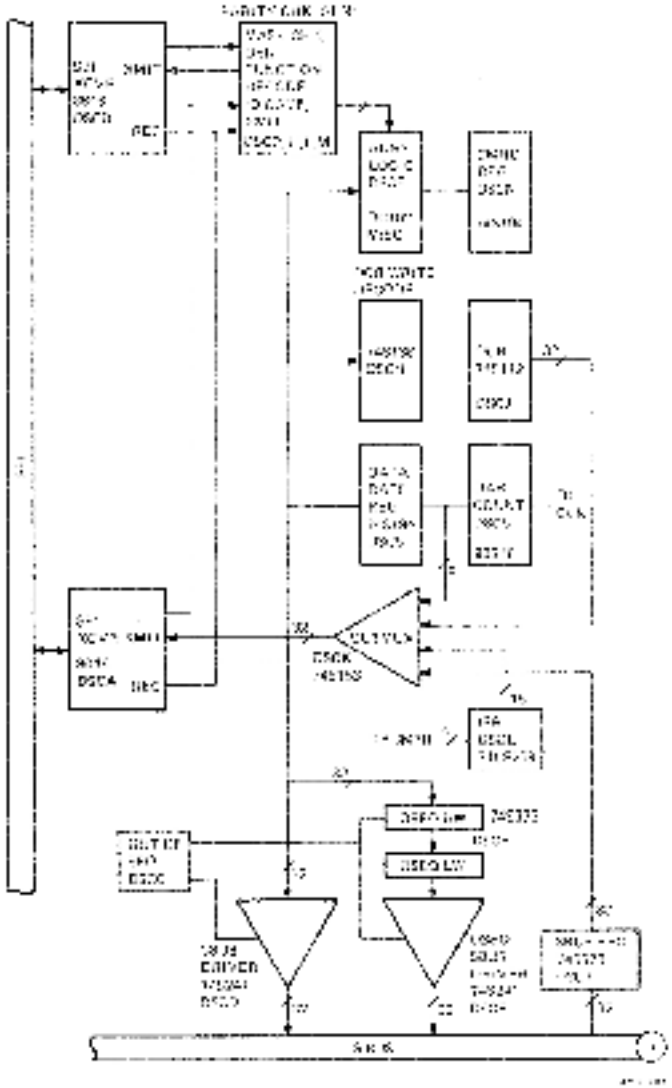


Figure 16-7 88/80 Block Diagram (Sheet 1 of 4)

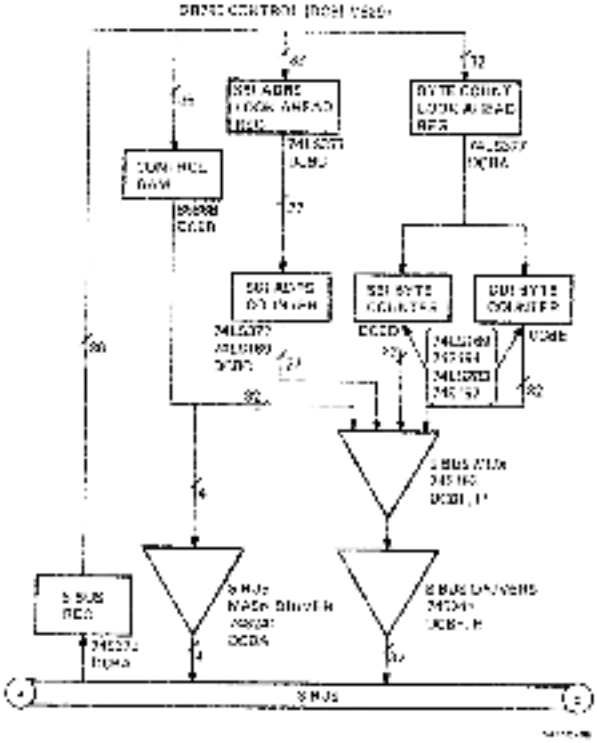


Figure 16-7 DR700 Block Diagram (Sheet 2 of 4)

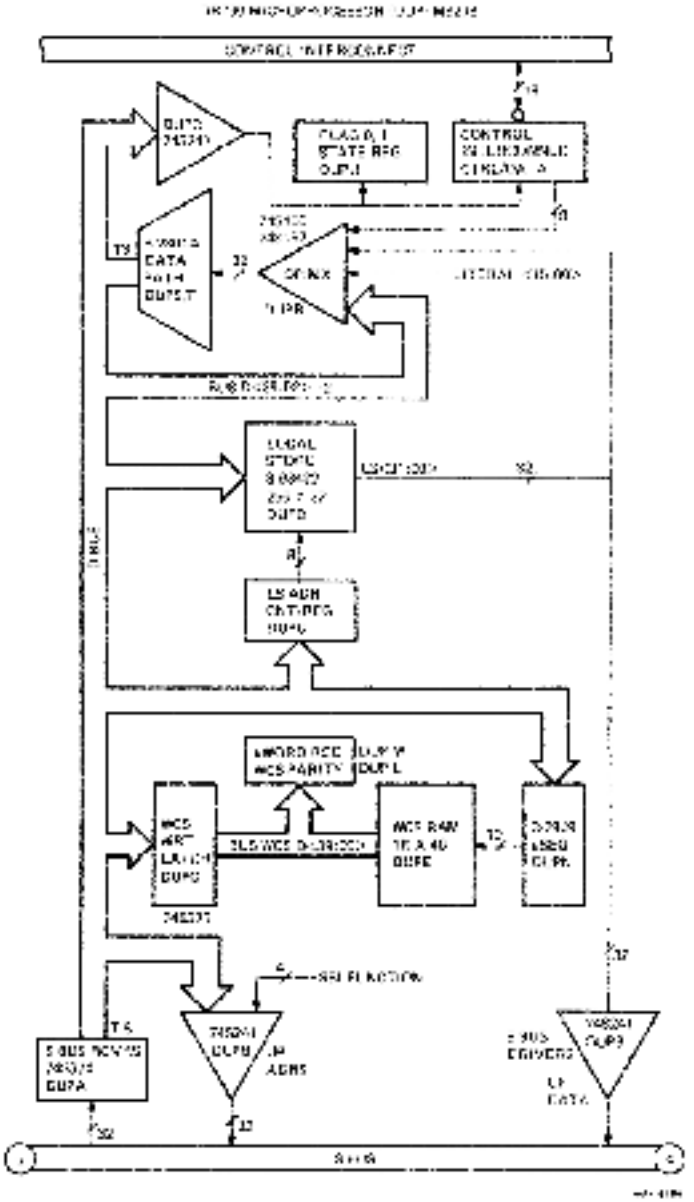


Figure 16-7 DR780 block Diagram (Sheet 3 of 4)

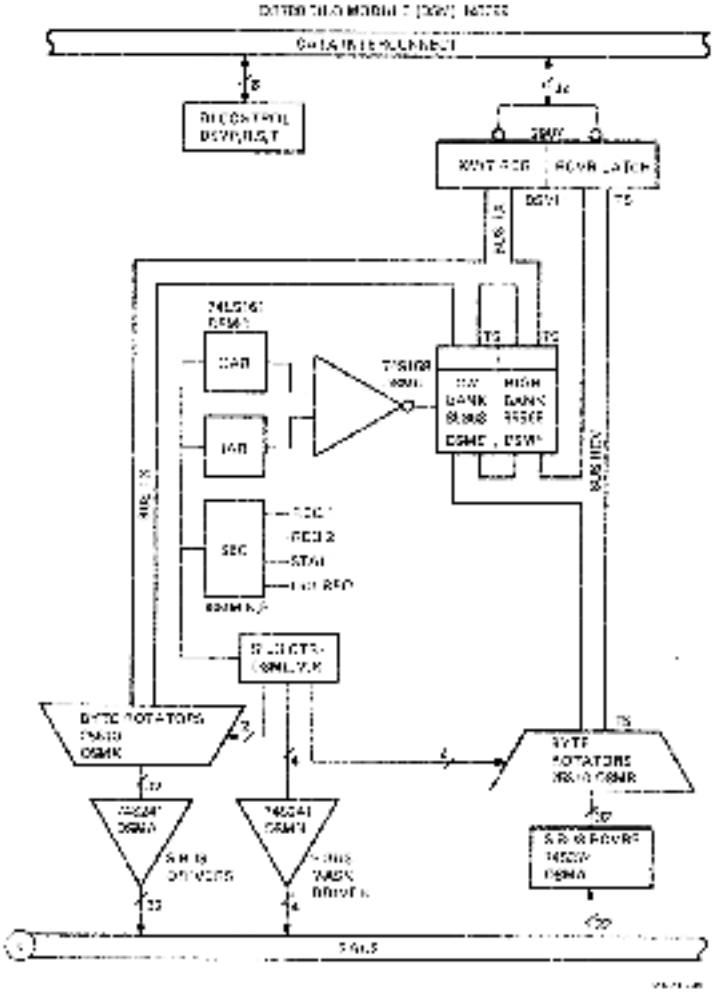


Figure 16-7 [illegible] Block Diagram (Sheet 4 of 4)

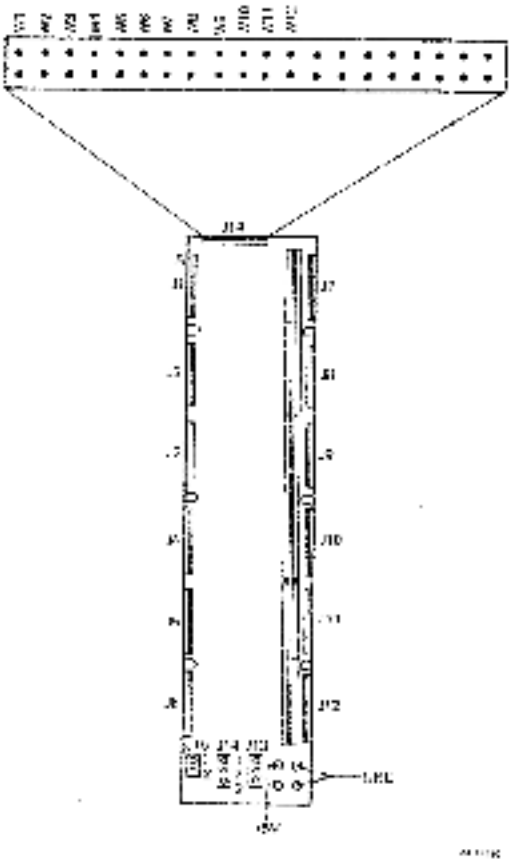


Figure 16-8 DS700 Backplane Jumper Location

Table 16-5 DR780 Backplane Jumper Settings and Wirewrap Selection

DR No.	DA SELD L	DR SELC F	DR SEL8 K	DR SELA I	Wirewrap P02L2 To
1	--	--	--	--	P02C1
2	--	--	--	I	P02T1
3	--	.	I	--	P02S1
4	--	--	I	I	P02P2
5	--	F	--	--	P02E2
6	--	I	--	I	P02J1
7	--	I	I	--	P02J2
8	--	I	I	I	P02M1
9	.	.	--	--	P02N1
10	I	--	--	I	P02P1
11	I	--	I	--	P02P2
12	I	--	I	I	P02E2
13	I	I	--	--	P02T2
14	F	I	--	I	P02U1
15	I	I	I	--	P02C2
16	I	I	I	I	--

P02E8

1. An 'I' in the table indicates that a jumper is installed.

2. The first DR780 is at TR 11, and has W4 and W9 installed. The second DR780 is at TR 11 and has W4, W3, and W1 installed.

3. The first DR780 has a wirewrap from P02L2 to P02T2. The second DR780 has a wirewrap from P02L2 to P02U1

4. W5, W6: For a BR of 6, W5 is in and W6 is out.

5. W7, MSYN Jumper Selects: If the DR780 is not going to perform bus arbitration, or be its master, then W7 is installed. If the DR780 is to be the master device, the jumper is not installed.

6. W8, BT Clock Jumper Selects: If the DR780 is to be the clock source for the Data Interconnect (DI), then jumper W8 is not installed. If the Customers device is to be the clock source for the DR780 device Interconnect (DVI), then jumper W8 is installed.

7. W9 - W12: These jumpers are not used by the DR780, but may contain spare jumpers for installation in W8 or W7.

8. Diagnostic Wirewrap Selection: If the DR780 interface is device A, wirewrap E03R1 to E03T1 and E03R2 to E03P2. Otherwise, wirewrap E03R1 to E03P2 and E03R2 to E03T2. When connecting two DR780 interfaces (DR780 to DR780), only one can be device A.

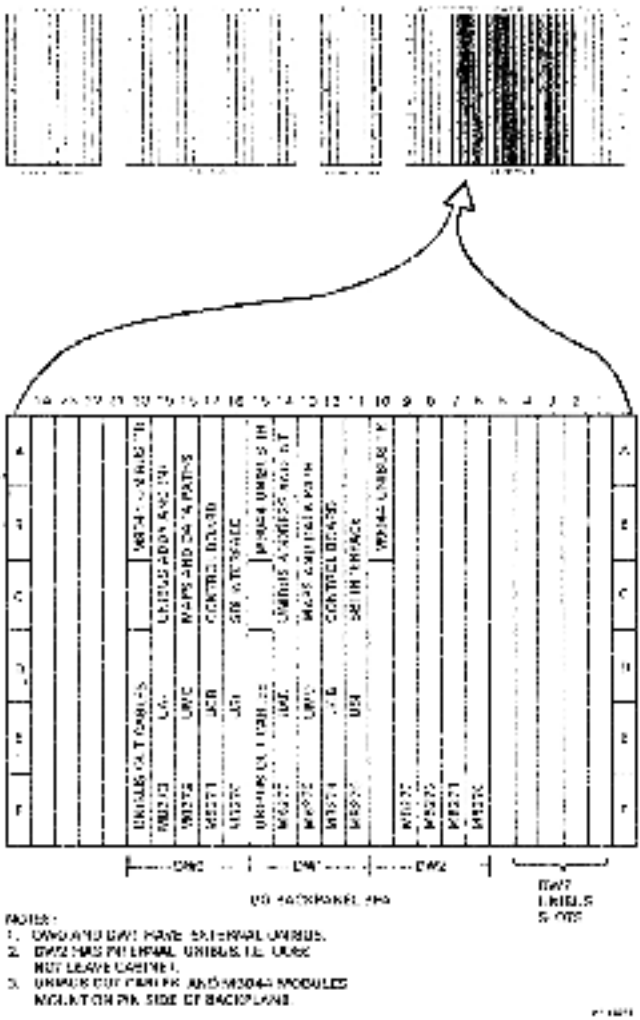


Figure 16-9 DW780 I/O Backplane Module Utilisation



Figure 16-10 DW780 Expander Cabinet Module Utilization

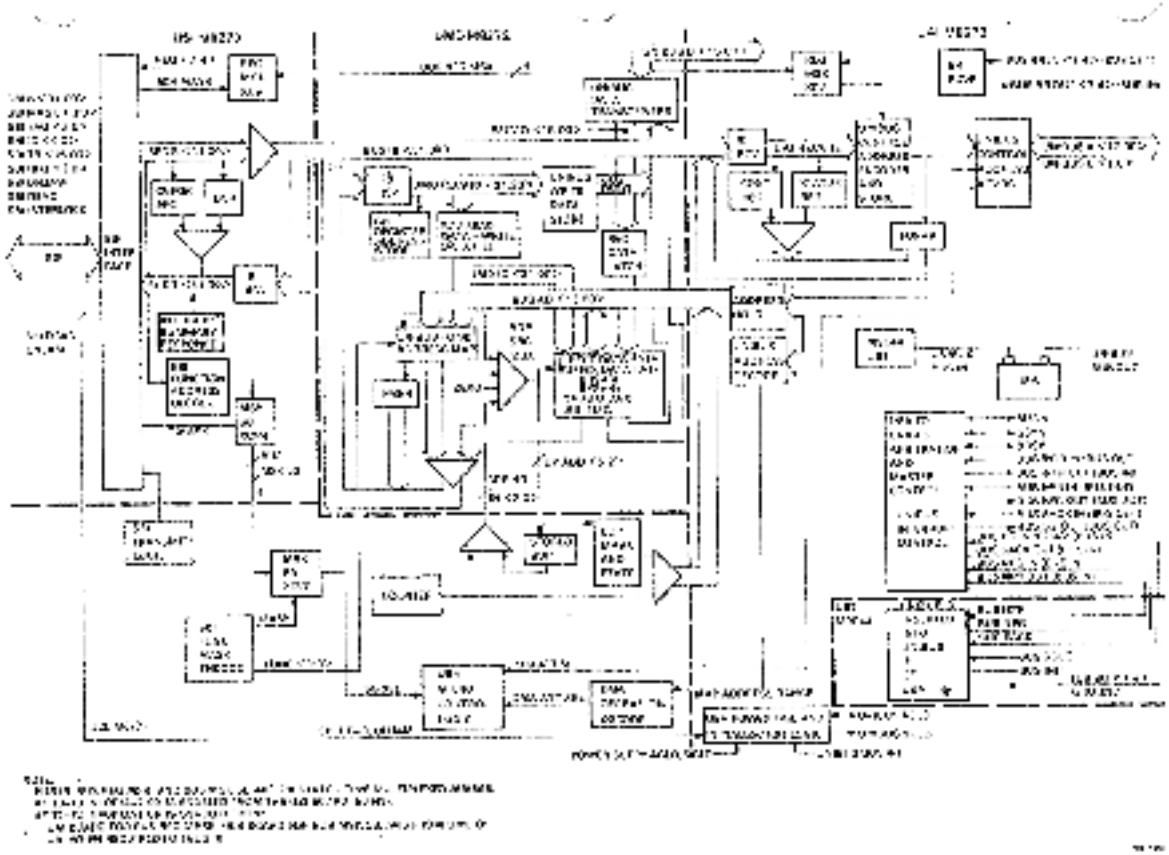


Figure 16-11 DM780 Block Diagram

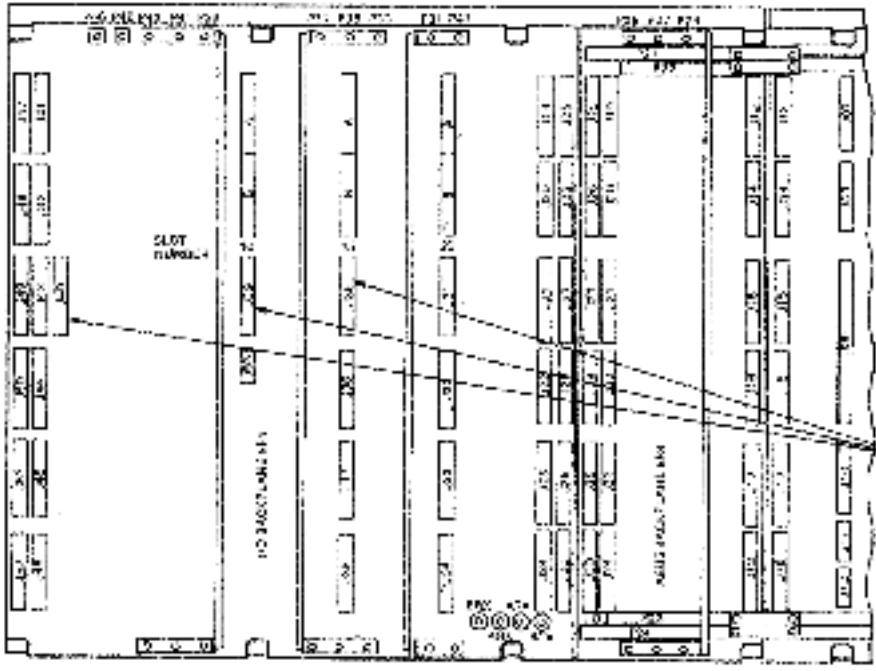


Figure 16-12 DM760 I/O Backplane Jumper Locations

16.4.1 DW780 Jumper Settings

The DW780 jumpers are installed in the I/O backplane, in J35, J39, and J40 for DW0, DW1, and DW2 respectively. If the DW780 is installed in the SBI expansion cabinet, the jumpers are installed in J24 of the UBA backplane.

The DW780 jumpers in the I/O backplane are not the same as the DW780 jumpers in an SBI expansion cabinet. Although Unibus adapter (UBA) interrupt level selection is the same, UBA TR arbitration selection differs by slot numbers, and the jumpers for UBA Unibus address space selection are reversed. Table 16-6 lists the DW780 unibus adapter jumpers.

Table 16-6 DW780 Jumper Settings

UBA TR Arbitration Level Selection

TR No.	USIC TX SEL A L	USIC TR SEL B L	CSIC TR SEL C L	SBIC TR SEL D L	Wirewrap From DXXR2 - To
	W3	W4	W5	W6	
1	--	--	--	--	FXXCJ
2	1	--	--	--	FXXDJ
3	--	1	--	--	FXXEJ
4	1	1	--	--	FXXFJ
5	--	--	1	--	FXXHJ
6	1	--	1	--	FXXJJ
7	--	1	1	--	FXXKJ
8	1	1	1	--	FXXLJ
9	--	--	--	1	FXXMJ
10	1	--	--	1	FXXPJ
11	--	1	--	1	FXXRJ
12	1	1	--	1	FXXSJ
13	--	--	1	1	FXXTJ
14	1	--	1	1	FXXUJ
15	--	1	1	1	FXXVJ
16	1	1	1	1	--

NOTES

1. The "1" indicates that the jumper is installed to provide a low (true) signal.

2. The XX indicates the slot number as follows.

   a. I/O backplane DW0 - I/O backplane slot 10

   b. I/O backplane DW1 - I/O backplane slot 11

   c. I/O backplane DW2 - I/O backplane slot 8

   d. SBI expansion cabinet - UBA backplane Slot 01

3. The first UBA (DW0) is jumpered as TR3. Subsequent UBAs are jumpered as TR4, TR5, and TR6.

Table 16-6 DW730 Jumper Settings (Cont'd)

CBA Unibus Address Space Selection

SBI Expansion Cabinet

Adapter Number	OSID Adapter 1 L W1	OSID Adapter 0 L W2
0	--	--
1	--	I
2	I	--
3	I	I

I/O Backplane

Adapter Number	BSID Adapter 1 L W7	BSID Adapter 0 L W1
0	--	--
1	--	I
2	I	--
3	I	I

WARNING

The use of W1 and W2 in the expansion cabinet has the opposite use as in the I/O cabinet.

NOTE

The "I" indicates that the jumper is installed to provide a low (true) signal.

CBA Interrupt Level Selection

Request Level	UAIF SBI PRI JMP 0 L W7	UAIr SBI PRI JMP 1 L W8
4	--	--
5	I	--
6	--	I
7	I	I

NOTES

1. The "I" indicates that the jumper is installed to provide a low (true) signal.
2. The UBAs are installed as UBA devices.

16.4.2 UDA 50 Module Utilization

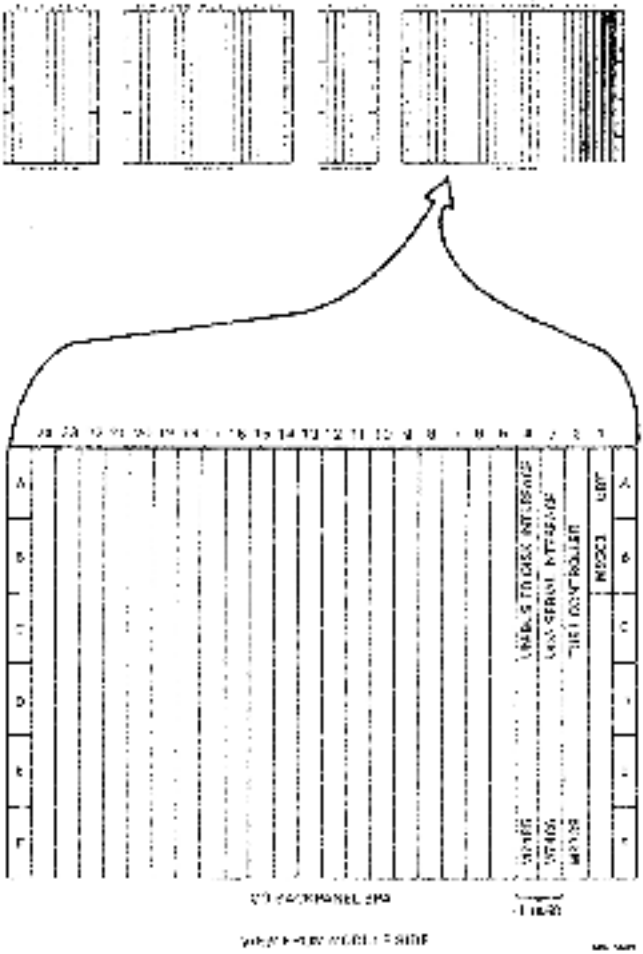


Figure 16-13 UDA 50 Module Utilization

16.4.3 Unibus Signals

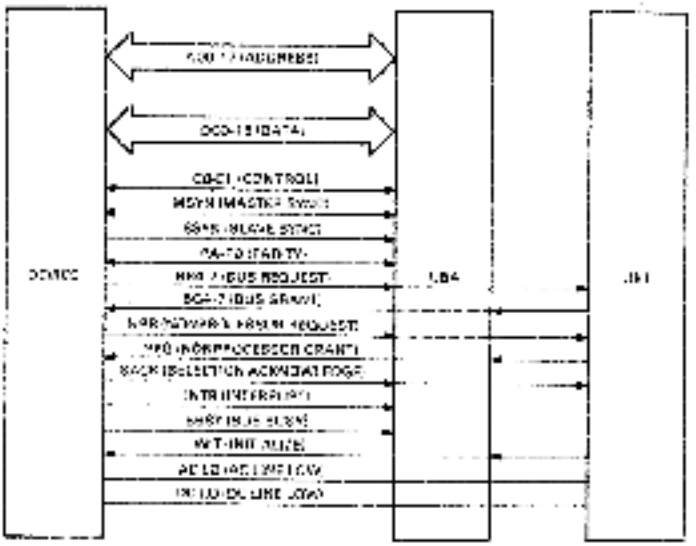


Figure 16-14 Unibus Signals

Table 16-7 Unibus Signal Descriptions

Signal	Description
Data Transfer Group	
Address lines SA<17:00>	These lines are used by the master device to select the slave (a unique memory or device register address). SA<17:01> specifies a unique 16-bit word. SA<00> specifies a byte within the word.
Data Lines [D<15:00>]	These lines transfer 16 bits of information between master and slave.
Control [C1, C0]	These signals are coded by the master device to control the slave in one of four possible data transfer operations as specified below. The transfer direction is always designated with respect to the master device.

Table 16-7 Unibus Signal Descriptions (Cont.)

Signal	Description
	C1 C0
	0 0 Data In (DATI): A data word or byte is transferred into the master from the slave.
	0 1 Data In Pause (DATIP): Similar to DATI except that it is always followed by a DATO(B) to the same location.
	1 0 Data Out (DATO): A data word is transferred out of the master to the slave.
	1 1 Data Out Byte (DATOB): Identical to the DATO except a byte is transferred instead of a full word.
Parity (PA, PB)	These signals transfer Unibus parity information. PA is currently unused and not asserted. PB, when true indicates a device parity error.
Master Synchronization (MSYN)	MSYN is asserted by the master to indicate to the slave that valid address and control information (and data for a DATO or DATOB) is present on the bus.
Slave Synchronization (SSYN)	SSYN is asserted by the slave. On a DATO it indicates that the slave has latched the write data. On a DATI(P) it indicates that the slave has asserted read data on the Unibus.
Interrupt (INTR)	This signal is asserted by an interrupting device after it becomes bus master to inform the UBA that an interrupt is to be performed, and that the interrupt vector is present on the D lines. INTR is negated upon receipt of the assertion of SSYN by the UBA at the end of the transaction. INTR may be asserted only by a device that obtained bus mastership under the authority of a BG signal.
Priority Arbitration Group	
Bus Request (BR7-BR4)	These signals are used by peripheral devices to request control of the bus for an interrupt operation.
Bus Grant (BG7-BG4)	These signals form the CPU and UBA response to a bus request. Only one of the four will be asserted at any one time.

Table 16-7 Unibus Signal Descriptions (Cont.)

Signal	Description
Nonprocessor Request (NPR)	This is a bus request from a device for a transfer not requiring CPU intervention (i.e., DMA).
Nonprocessor Grant (NPG)	This is the grant in response to an NPR.
Selection Acknowledge (SACK)	SACK is asserted by a bus-requesting device after having received a grant. Bus control passes to this device when the current bus master completes its operation.
Bus Busy (BBSY)	BBSY indicates that the data lines of the bus are in use. It is asserted by the Unibus master.
Initialization Group	
Initialize (INIT)	Asserted by the terminator board (slot 9). Each DC LO is asserted on the Unibus. It stays asserted for 10 milliseconds following the negation of DC LO.
AC Line Low (AC LO)	This is an anticipatory signal that warns of an impending power failure. AC LO initiates the power fail trap sequence and may also be issued to peripheral devices to terminate operations in preparation for power loss.
DC Line Low (DC LO)	This signal is available from each system power supply and remains clear as long as all dc voltages are within the specified limits. If an out-of-voltage condition occurs, DC LO is asserted.

Table 16-8 Unibus Pin Assignments - Standard and Modified

Pin	Standard Signal	Modified Signal	Pin	Standard Signal	Modified Signal
AA1	INIT L	INIT L	DA1	D<6 H	SPARE
AA2	+5 V	+5 V	DA2	+5 V	+5 V
BA1	INTR L	INTR L	DB1	D<5 H	SPARE
BA2	GROUND	TEST POINT	DB2	GROUND	TEST POINT
CA1	D00 L	D00 L	DC1	D<4 L	D04 L
CA2	GROUND	GROUND	DC2	GROUND	GROUND
DA1	D01 L	D01 L	DD1	GROUND	BAT BACKUP +5 V
DA2	D02 L	D02 L	DD2	D<4 L	D04 L
EA1	D04 L	D04 L	DE1	GROUND	TMT SSYN*
EA2	D05 L	D05 L	DE2	D<4 H	PSEUDO*
FA1	D06 L	D06 L	DF1	AC LO L	AC LO L
FA2	D07 L	D07 L	DF2	DC LO L	DC LO L
HA1	D08 L	D08 L	DH1	AD1 L	AD1 L
HA2	D07 L	D07 L	DH2	AD0 L	AD0 L
JA1	D10 L	D10 L	DJ1	AD2 L	AD2 L
JA2	D09 L	D09 L	DJ2	AD3 L	AD3 L
KA1	D12 L	D12 L	DK1	AD5 L	AD5 L
KA2	D11 L	D11 L	DK2	AD4 L	AD4 L
LA1	D14 L	D14 L	DL1	AD7 L	AD7 L
LA2	D13 L	D13 L	DL2	AD6 L	AD6 L
MA1	PA L	PA L	DM1	AD9 L	AD9 L
MA2	D15 L	D15 L	DM2	AD8 L	AD8 L
NA1	GROUND	P1*	DN1	A11 L	A11 L
NA2	PB L	PB L	DN2	A10 L	A10 L
PA1	GROUND	P0*	DP1	A13 L	A13 L
PA2	BUSY L	BUSY L	DP2	A12 L	A12 L
RA1	GROUND	BAT BACKUP +5 V	DR1	A15 L	A15 L
RA2	SACK L	SACK L	DR2	A14 L	A14 L
SA1	GROUND	BAT BACKUP +5 V	DS1	A17 L	A17 L
TA1	NPR L	NPR L	DS2	A16 L	A16 L
TA2	GROUND	GROUND	DT1	GROUND	GROUND
UA1	BR7 L	BR7 L	DT2	C1 L	C1 L
UA2	NPG H	+20 V	DU1	SSYN L	SSYN L
VA1	BG5 L	BG5 L	DU2	C0 L	C0 L
VA2	BG7 H	+20 V	DV1	MSYN L	MSYN L
WA1	GROUND	120 V	DV2	GROUND	-5 V

*Pins used by parity control module.

16.1.4 Unibus Address to VAX Physical Address Conversion

An example will be used to show how to convert from a Unibus byte address (Octal) of 760270 to the VAX physical address (Hex).

Example 16-3 Conversion from Unibus address to VAX physical address

1. First change the octal number to binary digits.

   111 110 000 010 111 000

2. Proceed to convert the binary number into hexadecimal by re-ordering the octal binary number into 4-digit segments.

   11 1110 0000 1011 1000

3. Add the appropriate DW780 Unibus adapter Base Address to the HEX number of the preceding step (Figure 16-4). For this example we will use UBA, 20 000000. The resultant number is the HEX representation of the VAX Physical Byte Unibus Address.

```
20100000        Unibus Adapter Base Address (Adapter 0)
+  3E0B8        HEX representation of Unibus address
--------
201E00B8        HEX representation of the Unibus address
                converted to a VAX Physical Byte Unibus
                address
```

NOTE

For a word access, always examine on a word boundary.

16.1.5 VAX Physical Address (Hex) to Unibus Address (Octal) Conversion

An example will be used to show the conversion from VAX physical address space to Unibus address space. The physical address of 20.1E3002 will be used.

1. To make this conversion, the physical byte address must be assigned to the Unibus space. For the DW780 adapter, See Figure 16-4 to compare the address with the range of addresses for each UBA. Determine which UBA corresponds to this address.

2. Change the HEX digits to the binary equivalent and express how is least significant bits (bits 17-0).

```
          2     0     1     E     3     0     0     2

        0010 0000 0001 1110 0011 0000 0000 0010
        ---------------------------------------
        31                17                    0

                    11 1100 0110 0000 0000 1000
```

3. Reformat the 18 binary bits into 3-bit sections. The result is the unibus byte address in octal.

(1) 110 000 000 001 000 = 760010

16.4.6 DW780 Unibus Device Addresses

Table 16-9 is a list of VAX physical addresses for unibus devices.

Table 16-9 DW780 Unibus Device Addresses

Unibus Device Address	Equivelent DW780 Adapter "HSTX" Address (SBIA 0)			
	#0	#1	#2	#3
760010	2013E008	2017E008	2018E008	201FE008
760020	2013E010	2017E010	2018E010	201FE010
760030	2013E018	2017E018	2018E018	201FE018
760040	2013E020	2017E020	2018E020	201FE020
760050	2013E028	2017E028	2018E028	201FE028
760060	2013E030	2017E030	2018E030	201FE030
760070	2013E038	2017E038	2018E038	201FE038
760100	2013E040	2017E040	2018E040	201FE040
760110	2013E048	2017E048	2018E048	201FE048
760120	2013E050	2017E050	2018E050	201FE050
760130	2013E058	2017E058	2018E058	201FE058
760140	2013E060	2017E060	2018E060	201FE060
760150	2013E068	2017E068	2018E068	201FE068
760160	2013E070	2017E070	2018E070	201FE070
760170	2013E078	2017E078	2018E078	201FE078
760200	2013E080	2017E080	2018E080	201FE080
760210	2013E088	2017E088	2018E088	201FE088
760220	2013E090	2017E090	2018E090	201FE090
760230	2013E098	2017E098	2018E098	201FE098
760240	2013E0A0	2017E0A0	2018E0A0	201FE0A0
760250	2013E0A8	2017E0A8	2018E0A8	201FE0A8
760260	2013E0B0	2017E0B0	2018E0B0	201FE0B0
760270	2013E0B8	2017E0B8	2018E0B8	201FE0B8
760300	2013E0C0	2017E0C0	2018E0C0	201FE0C0
760310	2013E0C8	2017E0C8	2018E0C8	201FE0C8
760320	2013E0D0	2017E0D0	2018E0D0	201FE0D0
760330	2013E0D8	2017E0D8	2018E0D8	201FE0D8
760340	2013E0E0	2017E0E0	2018E0E0	201FE0E0
760350	2013E0E8	2017E0E8	2018E0E8	201FE0E8
760360	2013E0F0	2017E0F0	2018E0F0	201FE0F0
760370	2013E0F8	2017E0F8	2018E0F8	201FE0F8
760400	2013E100	2017E100	2018E100	201FE100
760410	2013E108	2017E108	2018E108	201FE108
760420	2013E110	2017E110	2018E110	201FE110
760430	2013E118	2017E118	2018E118	201FE118

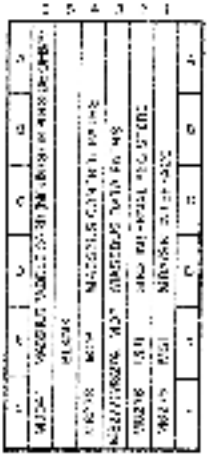
Table 6-4 DW780 Unibus Device Address (Cont.)

Unibus Device Address	Equivalent DW780 Adapter "BYTE" Address (SBIA 0) #0	#1	#2	#3
760010	2013E100	2017E100	201BE100	201FE100
760450	2013E128	2017E128	201BE128	201FE128
764004	2013E804	2017E804	201BE804	201FE80A
764014	2013E80C	2017E80C	201BE80C	201FE80C
764024	2013E814	2017E814	201BE814	201FE814
772460	2013F130	2017F130	201BF130	201FF130
772410	2013F108	2017F508	201BF108	201FF508
774420	2013F900	2017F900	201BF900	201FF900
777160	2013FE70	2017FE70	201BFE70	201FFE70
777440	2013FF20	2017FF20	201BFF20	201FFF20
777514	2013FF40	2017FF40	201BFF40	201FFF40

NOTE

For DW780 addresses on SBIA 1, simply add 02000000 to the address. For example, Unibus address 760010 on DW0, SBIA 1 will be 221FE010.

16.5 8R780



NOTES:
1. REVISION NUMBER = 21xx
2. THE MODEL NUMBER PARTIAL CARD MOUNTED ON THE BACK-PLANE SLOT IS EMPTY.
3. THE 8R783 MEMORY IS SHIPPED AND DOES CONTAINS 4 32 K BYTES.
4. THE 8R784 MEMORY MODULE HAS AN EIGHTY-SIX (86) PIN CONNECTOR AND IS USED FOR HIGHER SPEED CLOCKS.

Figure 16-15 8R780 Module Identification

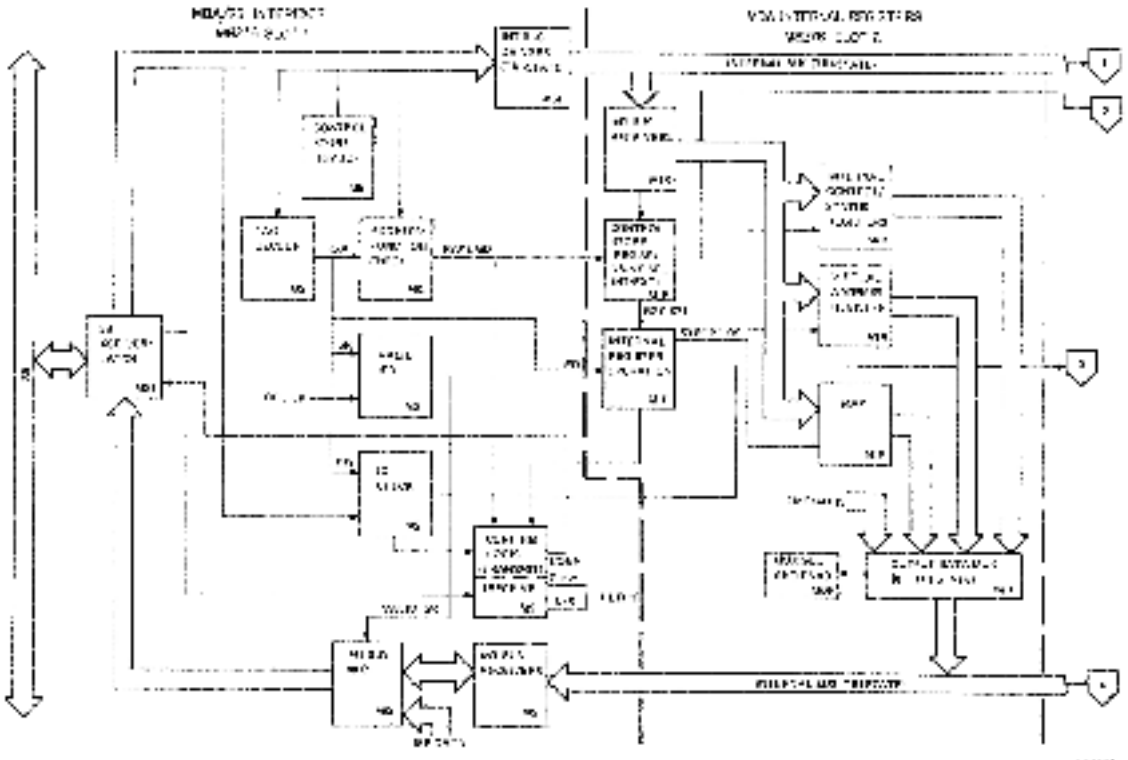


Figure 16-16 RH702 Block Diagram (Sheet 1 of 2)

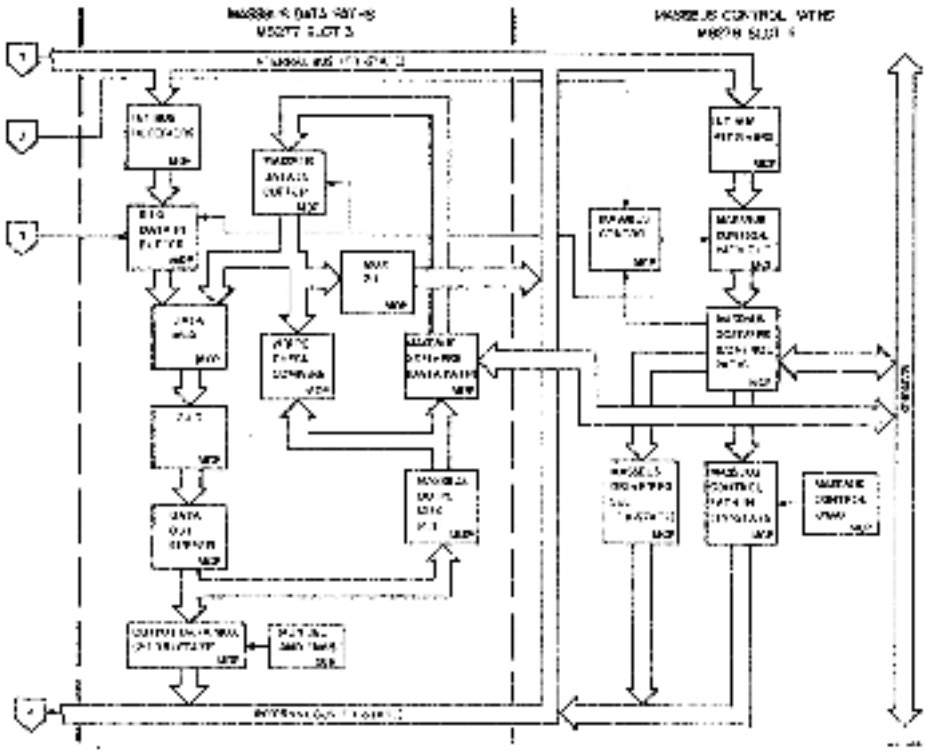


Figure 16-15 RR780 Block Diagram (Sheet 2 of 2)

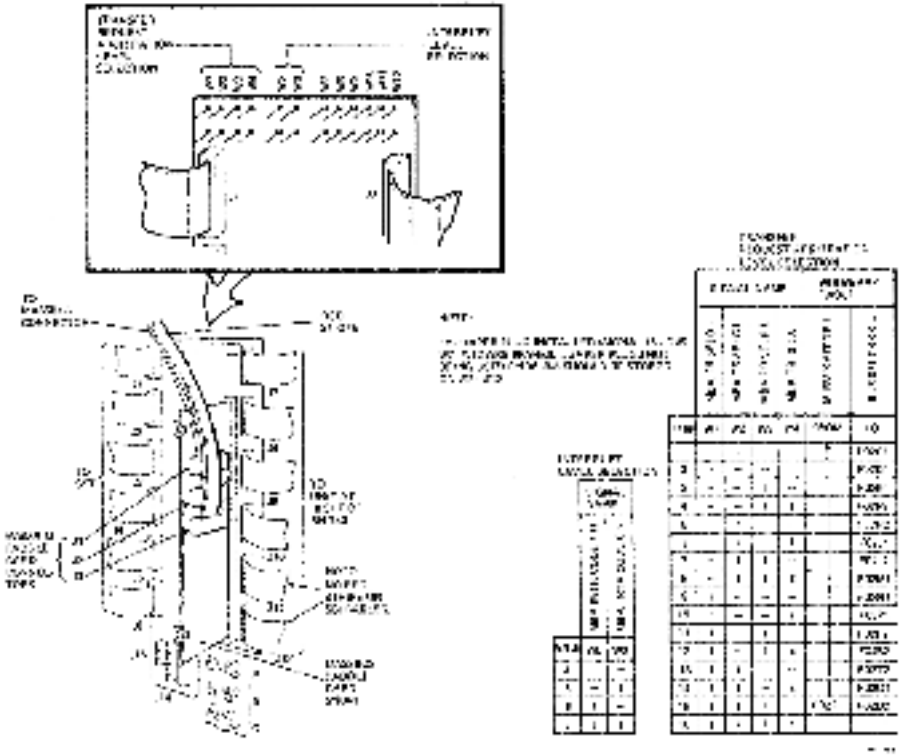


Figure 10-17 XB700 Jumper Settings

16.5.1 MAP Register Address Calculation

The following steps may be used to calculate a MAP register address.

1. Determine the desired MAP Register location or offset. It must be in the range of 00-FF (Hex) as there are 256 MAP Registers in the MBA.

2. Enter the MAP Register offset into the MAP Register Address Calculation diagram (Figure 16-18). The register offset will be shifted two bits left, creating a longword boundary.

3. Add the MBA base address (Table 16-10) to the calculated address. The result is the CPU HEX address of the desired MAP register.

MAP REGISTER ADDRESS CALCULATION

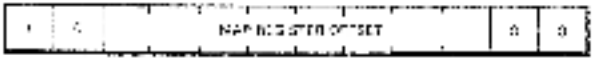


Figure 16-18 Map Register Address Calculation

16.5.2 MASSBUS Register Address Calculation

The following steps may be used to calculate a MASSBUS register address. The example will be used to calculate the MASSBUS register address for the control and status register (register 0) for drive 7, SBI 1, RH780 # 0.

1. Use the MASSBUS Register Offset Table (Table 16-11) and determine the correct offset for the desired register.

2. Add this offset to the MBA Base Address from Table 16-10. The result is the CPU HEX address of the desired external register. Do not let the bit configuration of Figure 16-19 confuse you. Bits 01 and 00 will always be zero, bits <06:02> indicate which register is being selected, and bits <09:07> indicate which drive is selected.

Example 16-4 Calculating a physical byte address

Calculate the physical byte address of CS1 (register 0) for drive 7, SBI 1, RH780 # 0 (TR3).

1. MBA base address from Table 16-10 = 22010000

2. MASSBUS register offset from Table 16-11 = 380

3. Adding the two together provides the physical byte address of 22010380.

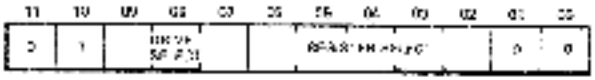


Figure 16-10 MASSBUS Register Address Calculation

Table 16-10 MBA Base Addresses

IP Level	SBIA 0 Base Address	SBIA 1 Base Address
0	20010000	22010000
9	20012000	22012000
10	20014000	22014000
11	20016000	22016000

NOTE

Always examine MASSBUS registers on a longword boundary.

Table 16-11 MASSBUS Register Offset

MBA NO.		MASSBUS DEVICE TYPE				DRIVE NUMBER							
HEX	OCT	RP DISK	RM DISK	TE TAPE	TM78 TAPE	0	1	2	3	4	5	6	7
0	0	CS1	RMCS1	CS1	CAS00	0	80	100	180	200	280	300	380
1	1	DS	RMDS	DS	CAS01	4	84	104	184	204	284	304	384
2	2	ER1	RMER1	ER	CAS02	8	88	108	188	208	288	308	388
3	3	MR	RMMR1	MR	CAS03	C	8C	10C	18C	20C	28C	30C	38C
4	4	AS	RMAS	AS	CAS04	10	90	110	190	210	290	310	390
5	5	DA	RMDA	FC	CAS05	14	94	114	194	214	294	314	394
6	6	DT	RMDT	DT	CAS06	18	98	118	198	218	298	318	398
7	7	LA	RMLA	CK	CAS07	1C	9C	11C	19C	21C	29C	31C	39C
8	10	SN	RMSN	SN	CAS08	20	A0	120	1A0	220	2A0	320	3A0
9	11	OFC	RMOF	TC	CAS09	24	A4	124	1A4	224	2A4	324	3A4
A	12	DCA	RMCC		CAS10	28	A8	128	1A8	228	2A8	328	3A8
B	13	MPA	RMMR		CAS11	2C	AC	12C	1AC	22C	2AC	32C	3AC
C	14	ER2	RMER2		CAS12	30	B0	130	1B0	230	2B0	330	3B0
D	15	EC3	RMEC2		CAS13	34	B4	134	1B4	234	2B4	334	3B4
E	16	CCCPOS	RMEC1		CAS14	38	B8	138	1B8	238	2B8	338	3B8
F	17	BCOFAT	RMOF1		CAS15	3C	BC	13C	1BC	23C	2BC	33C	3BC
10	20				CAS16	40	C0	140	1C0	240	2C0	340	3C0
11	21				CAS17	44	C4	144	1C4	244	2C4	344	3C4
						•	•	•	•	•	•	•	•
1F	37					7C	FC	17C	1FC	27C	2FC	37C	3FC

CHAPTER 17

REVISION CONTROL

17.1 VAX 8600/8650 REVISION INFORMATION

The following tables are a check list for the EA86 unit revision. They do not replace the SN document and are to be used for reference only. They are a history of changes that have occurred to the EA86 from revision B4 on K5. For prior history, see early revisions of the SN document. The following information is subject to change.

Table 17-1 shows revision information for the VAX 8600 and Table 17-2 shows revision information for the VAX 8650. Table 17-3 and Table 17-4 contain the diagnostic media revision information for VAX 8600 and VAX 8650 respectively.

For file revision information, read the file EA86RV.MEM.

Table 17-1 VAX MACRO Register Privileges

				PRIVILEGE REVISION		
UNIT REVISION LEVEL COMPATIBILITY>		01	01	02,03	04	05
Name	Description			Exclusions		
MACRO Library						
72-12193-01	MCH $ACTAPL=MC	01,02	01,02	01,02	01,02	01,02
72-12194-01	I/O $ACTIL=MC	01,00	01,00	01,00	01,00	01,00
72-12195-01	SMG$ MACRO $ASN	01,02	01,02	01,02	01,02	01,02
72-12196-01	CTL $ACTPL=MC	01,02	01,02	01,02	01,02	01,02
Routines						
L0220-CA	MEMORY MGMT (VA MGT, PAGE MGT	01,00	01,00	01,01	01,01	01,01
L0221-03	SYSCL SYMBOLIC	01,04,05,06,07	01,02,03,04,05,06,07	01,02,03,04,05,06,07	01,02,03,04,05,06,07	01,02,03,04,05,06,07
L0277-01	SMG$ INIT	01,02,03	01,02,03	01,02,03	01,02,03	01,02,03
L0289-03	SMG$ IDCA MBUS INTERFACE	01,02,03,04,05 Note 1	01,02,03,04,05 Note 1	01,02,03,04,05 Note 1	01,02,03,04,05 Note 1	01,02,03,04,05 Note 1
L0304-03	SMG$ MBOX	00,05,06,07	05,06,07,08,09	05,06,07,08,09	06,07,08,09,0A	06,07,08,09,0A
L0305-03	SMG$ MSGX	02,04,05,06,07	03,04,05,06,07	04,05,06,07,08	05,06,07,08,09	05,06,07,08,09
L0306-03	SMG$ IMBX	03,04,05,06	04,05,06,07	05,06,07,08	06,07,08,09,0A	06,07,08,09,0A
L0307-02	SMG$ IDBX	01	01	01,02	01,02	01,02
L0308-01	SMG$ LNGX	04,05,06,07	06,07,08,09,0A	07,08,09,0A,0B	08,09,0A,0B,0C	08,09,0A,0B,0C
L0309-01	SMG$ PBDX	00,01,02,03	01,02,03,04	02,03,04,05	03,04,05,06	03,04,05,06
L0310-02	SMG$ EDCX	03,04,05,06,07	04,05,06,07,08	05,06,07,08,09	06,07,08,09,0A	06,07,08,09,0A
L0311-02	SMG$ KBDX	00,04,06,07	00,05,06,07	05,06,07,08	07,08,09,0A	07,08,09,0A
L0312-02	SMG$ FBDX	00,01,02,03,04	00,01,02,03,04	01,02,03,04,05	02,03,04,05,06	02,03,04,05,06
L0313-01	SMG$ FMDX	00,01,04 (Note 2)	01,02,03 (Note 2)	02,03,04,05 (Note 2)	03,04,05,06 (Note 2)	03,04,05,06 (Note 2)
L0314-00	SMG$ TBDX	01	01	01,02,03	02,03,04	02,03,04
L0315-00	SMG$ PADT	00,03,04	02,03,04	03,04,05	04,05,06	04,05,06
L0316-00	SMG$ LBDX	00,04,05,06	00,05,06,07	05,06,07,08	06,07,08,09	06,07,08,09
L0317-00	SMG$ CLDX	00,01,02	00,01,02	01,02,03	02,03,04	02,03,04
L0318-00	SMG$ PIDX	01	02	02	02	03
L0319-01	SMG$ LJDX	00,03,04	00,03,04	02,03,04,05	03,04,05,06	03,04,05,06
L0320-03	SMG$ MBOX	01,02,03,04	01,02	01,02	01,02	01,02
L0321-02	SMG$ NCHRS 'CHARACTER	01,02	01,02	01,02,03	01,02,04	01,02,03
L0322-01	SMG$ KEYBD	02	02	02	02	02
L0324-03	SMG$ TERMINATOR	04,05,06	04,05,06	04,05,06	04,05,07	04,05,06

Table 17-1 VAX M860 Revision Information (Cont.)

		M8-PAGE	M8-PAGE	M8-PAGE	M8-PAGE	M8-PAGE
	UNIT REVISION LEVEL COMPATIBILITY	J1	A1	A0,A1	A0	A1
PART NO.	Description	Revisions				
1007F5-00	MEMORY ARRAY (16 MEG 256Z-CA)	A1	A1	A1	A1	A1
1007J6-00	MEMORY ARRAY (8 MEG 128Z-CA)	A1,A2	A1,A2	A1,A2	A1,A2	A1,A2
1007K5-00	MEMORY ARRAY (32 MEG M8SA-CA)	A1	A1	A1	A1	A1
1007E0-00	MEMORY CARD 8Q+8Q	A1,A2,B1 (Note 3)	A1,A2,B1 (Note 3)	A1,A2,A1 (Note 3)	A1,A2,A1 (Note 3)	A1,A2,A1 (Note 3)
1007H2-	21/2SU CMOS PROCESSOR - M78HA-BA	B1	B1	B1	B1	B1,E1,E2 (Note 4)
1007J0-	21/2SU PACKET BUFFER	A1,A2	A1,A2	A1,A1	A1,A1	A1,A1
1007J2-	21/2SU DA A (A1)	B1	B1	B1	B1	B1
1007M1-	21/2SU SBI INTERFACE	C1,C1,B1	C1,C1,B1	C1,C1,B1	C1,C1,C1	C1,C1,C1
1007T0-	SMTED UNIBUS ADAPTER SGI	C,D	C,D	D,D	C,D	C,D
1007T1	UNIBUS ADAPTER CONTROL	C,D,E	C,D,E	B1,B2	B1,B2	B1,B2
1007J3-	2K/2SU IBUS MAP & BUFF	C,D	C,D	C,D	C,D	C,D
1007T3-	SMTED UNIBUS ADAPTER ADDRESS	D1	D1	D1	D1	D1
1008B3-	2K/2SU UNIBUS TERMINATOR PAR RAM	A	A	A	A	A
1008J7-	2/0 BUS/UNIBUS TERM	A1, (Note 5)	A2, (Note 5)	A2, (Note 5)	A1, (Note 5)	A1,(Note 6)
1008J4-	2K/2SU UNIBUS TERMINATOR	B,C	B,C	B,C	B,C	B,C
1008T0-	1 BY M780 F ADAPTER	D1	D1	D1	D	D1
1008J0-XA	UNIBUS DISK ADAPTER (UDA50-A)	H1,J2	H1,H2	H1,H2	H1,H2	H1,H2
1008J0-	UNIBUS DISK ADAPTER (UDA50-A)	H0	H0	H0	H0	H0
5-10500-AA	SMD (DA50/RC25 UDA50, M880 CA)	-	-	-	-	A0
50-19910-04	SMD (1/2 POWER BUILD-TEST MICRO, M880 CA)	A2	A2	A2	A2	A1
50-12010-04	SMD (OPTIMIZED MICRO, M880-BA)	A1	A1	A1	A1	A1

NOTES

1. Any field returned module at revision G must be upgraded to revision J.
2. Revision G8 cancelled.
3. Memory test revision are identical to prior A level (A1 or A2) memory micro, except the plot.
4. Revision E1 was out for test use only.
5. Four releases of the BM (current) spanned the M8940 to be at revision level J1 and C. This was incorrect. It should have been revision A1.

Table 17-2 VAX Data Revision Information

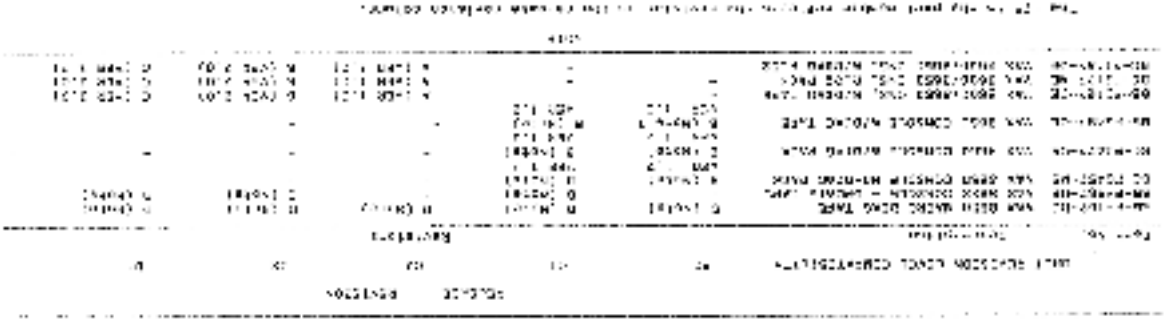
		RELEASE / REVISION				
UNIT REVISION LEVEL COMPATIBILITY		A?	F1	G?	C2	D
Part No.	Description	Revision				
Backplanes						
70-19111-01	BI0 BACKPLANE	A1,A2	A1,A2	A1,A2	A1,A2	A1,A?
70-19104-01	I/O BACKPLANE	C1,C2	C1,C2	D?,D?	C1,C2	C1,C2
70-19105-01	XBUS BACKPLANE	A1,A2	A1,A2	A1,A2	A?,A?	A1,A2
70-19106-01	I/O BACKPLANE	A1,A?	A1,A?	A1,A2	A?,A2	A1,A?
MS8600						
12??-??	MEMORY ARRAY (8 MEG, MS8600)	D?	D?	??	D1	D1
12??-??	LEDs	??=Unlit	F?	F?,G?,G1,H?	??,??,??,??	F1,F2,G1,G2
10??0-00	FSDS: SBIA	H?,H3,H4	G?,G2,G3	D1,D2,D3	H1,H?,H?	H1,H2,H3
10??0-??	CMPE: SBIA BUS INTERFACE	B1,B2,C?	H?,H4,G?,G3,C3	D1,D2,C?,C2,C3	?1,B3,C1,C2,C8	H1,H?,C?,C2,C3
		Note 1	Note 1	Note 1	Note 1	Note 1
10??04-??	(M:U) MMCR	F1	F1	F?,?2,?3	F1,F2,F3	F?,?2,F?
10??05-??	(M:?) MMCR	G1	G1	H?,H3,H4	G?,G3,G4,H4	C?,G2,G3,G4
10??06-??	(L:M) CSRS	H1	H1	H?,H?	H?,H3	H1,H3
10??07-??	(L:A) CSRS	H0,?0,??	H1	J1,J2	J?,J2	J?,J?
10??08-??	(CSR) IBCR	??,?? (??,??	??,?6,?7,?8	??,?5,?7,?8,?9	??,?8 ?? ??,??	??,?6,?7,?8,??
10??0?-??	(LU:) LUCR	G?,G8,G9,G?	G?,G3,G4,G5	G1,G3,G4,G5	G2 G8 G4 G5	G2,G3,G4,G5
10??10-??	(PRT) PMCR	C3,C4,C5,C6,C7	C4,C8 C9,C?,C?	C?,C4,C5,C6,C7,C8	C3 C4,C5,C6,C7,C8	C?,C4,C5,C6,C7,C8
10??1?-??	(SSD) SSCR	?1	E?	B1,B2	A? A3	B1,B2
10??1?-??	(MSR)	?1	H?,J2	J1,J2	J?,J2	F1,J?
10??1?-??	(MSU)	??,C3 (Note 2)	C?,C?,?? (Note 2)	D?,D?,D4 (Note 2)	D1,D?,D? (Note 2)	C?,E? (? (Note 2)
10??14-??	(CSR) EBCR	??,?5,?6,?7,?8	D1	H?,H2,H?	H1,H?,H4,H4	H1,H?,H4,H?
10??1?-??	(SSA) SSRR	B?,?8,??	A?,A5,A4	B2,B3,B4,B5	B2,B3,B4,B5	B?,B4 B4,B5
10??1?-??	(SSD) SSCR	B? B4,B5 ?8	B2,B3,B4,B5	B2,B3,B4,B5	B2,B3,B4,B5	B2,B3 B4,B5
10??1?-??	(S:M) SS TICR	A?	A2	A2	A2	A?
10??1?-??	(PRT) PMCR	B1,B3,B4	B?,B4,B5	B?,A?,B4,B5	B?,B3,B4,B5	B2 B4 B4,B5
10??2?-??	MEMORY TESTER/ATOR	B?,A?	B1 B2	B1,B?,B?	B1,B5,??	B1,B2,B3
10??5-??	(PTH) MT PRTR	A2	A2	B?	A2	A?
11??4-??	XBUS TERMINATOR	B4,B?,B4	B4,B5,B6	B4,B5,B6	A?,B? B?	B4,B5,B6
10??7-??	MMBMe Array (16 MEG, MS8?CA	A?	A1	A?	B1	B1
10??8-??	MEMORY ARRAY (4 MB), MS8? AA	A	A1,A2	B1,?2	A1,A2	A1,A?

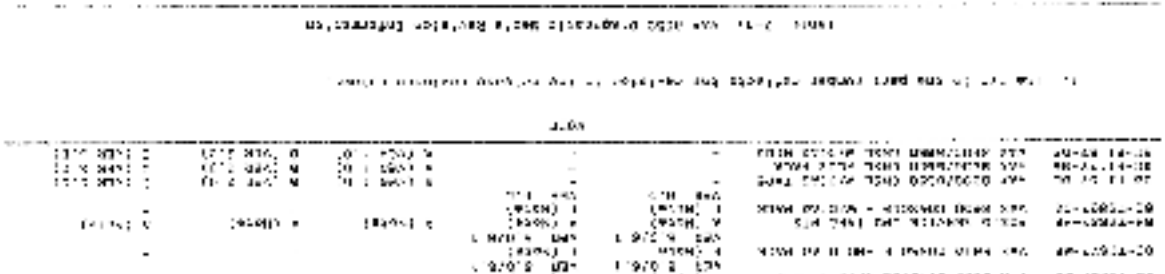
Table 1-2. Unit FRPO Revision Information (cont)

Unit Revision Level Compatibility	01	C1	C2	D1	D2
Part No.	Description	Revisions			
6320C-02	CPU: MEM	A1,A2	A1,A2	A1,A2	A1,A2	A1,A2
6321-02	CPU: CTRL	B1	B1	B1	B1	B1
6323-02	Memory Array (64 KB, Mem 0)	A1	A1	A1	A1	A1
6326-03	Memory Card Assy	A1,A2,B1 (Note 2)	A1,A2,C1 (Note 2)	A1,A2,C1 (Note 2)	A1,A2,D1 (Note 2)	A1,A2 B1 (Note 2)
1310*	CE Tool Interface Processor Interface	D1	D1	D	D1	D1,D1,C2 (Note 4,
1310	Disk Physical Unit	A1,A2	D1,D2	A1,C1	D1,D1	A1,A1
1310-	Disk Data Path	D1	D1	D1	D1	D1
1410*	CE Tool Interface	C1,C1,D1	C1,C1,D1	C,C,D1	C D1,D	C C1,D1
9832	Device Unique Adapter Set	D	D	D	D	D
9832-	Device Unique Adapter Control	C1,C2	C1,C2	C1,C2	C1,C2	C1,C2
9837*	Device Unique Adapt Emp & Cable	D	C	C	D	D
9837	Device Unique Adapter Set	D	C	C	D	D
9837*	Device Unique Terminator for Ins	A	A	A	A	A
9C040	I/O Unit Adapter Type	A2	A2(Note 3)	A2(Note 3)	A2(Note 3)	A2
9N047-	Diskette Unique Terminator	B,C	B,C	B,C	B,C	B,C
9R70*	TJA: Module Adapter	B1	C1	C1	C1	C1
9T08A-XA	Diskette Data Adapter Cable(-A)	A1	A1,A2	A1,A2	A1, A2	A1,A2
9T08A	Diskette Data Adapter Cable Assy	A2	A2	A2	A2	A2
5- 1501B-AA	SNA Terminator Kit(s), Mem-Ca)	A3	A3	A3	A3	A3
5A-18500-04	SNA LCA Power Unavailable Adapter, Model-04)	-	-	-	-	A1
5A-17257-AA	SNA Terminator Board, Mem Ma)	A1	A1	A1	A1	A1

Notes

1. Any field-required module as indicated must be upgraded to revision C.
2. Device is CE tool only.
3. Memory load modules are installed if units A and B of memory Mem 0 modules are A or C.
4. Device is B1 rev. of 1 for test only.
5. PU ECO issued by a minor change in revision of the 9C040 from A2 to B. A supplemental PUT procedure was issued changing the keyledge pack to A2.





CHAPTER 10

REMOVAL AND REPLACEMENT PROCEDURES

10.1 OVERVIEW

This chapter describes how to remove and replace assemblies in the kernel system. Only the major field replaceable units (FRUs) for the system are covered.

This chapter covers the two major system components included in a kernel system, the front and rear CPU cabinets, and is divided into four major parts, one part for each of these components, this introduction, and a section on module paddle connector cleaning.

The procedures are listed in a specific order because some procedures depend on the completion of preceding ones. Be sure you follow this order as you work on each system component, or you may run into problems in performing the procedures.

As you work on each component, pay attention to the following rules.

- Use all screws and locks when you reassemble a component and put them in their correct places.
- Do not bundle signal cables with power cables.
- Unless otherwise stated, to replace an FRU, reverse the order of the removal procedures.
- Perform the appropriate diagnostic test for each FRU that you replace.

18.2 FRU PART NUMBERS

		Digital PN
A.	FRUs FOR CPU CARRIER ASSEMBLY 60 Hz	70-19214-00
	50 Hz	70-19214-01
	POWER SUPPLY ASSEMBLY	70-20340-00
	(2) AC Input Assembly (2.8 Kw)	H7176-A
	(2) Modular Power Supply (+5 V @ 200 A)	H7180-A
	XMM Assembly (with PROM)	H7180-A
	(2) Modular Power Supply (+2.5 V @ 100 A)	H7187-A
	(2) Modular Power Supply (-5 V @ 20 A)	H7180-A
	Modules in CARD CAGE ASSEMBLY	FRUs available
	H7231 POWER SUPPLY 120 V/240 V	H7231-A
	876 POWER CONTROLLER 120 V/200 V	876-A
	BLOWER, CENTRIFUGAL 1500 CFM 60 Hz	12-19197-00
	50 Hz	12-19197-01
	Filler in CARD CAGE SUPPORT ASSEMBLY	74-20034-02
	Air Flow Sensor (with LED)	12-22605-01
	Temperature Sensor	12-19126-01
B.	FRUs FOR FRONT END CABINET ASSEMBLY 60 Hz	70-19213-00
	50 Hz	70-19213-01
	CVT	30-23959-00
	RL02 DISK DRIVE	RL02-PK
	BA11-A UNIT ASSEMBLY	BA11-AL
	Modules in BA11-A	FRUs available
	H7140 POWER SUPPLY in BA11-A	H7140
	FAN PANEL ASSEMBLY (50 Hz)	70-19805-00
	Filters (four, two in front door and two in back door)	12-11253-00
	Filler in Back Door Assembly	12-11255-02

18.3 FRONT END CABINET ASSEMBLY

This paragraph describes how to electrically disconnect the Front End cabinet from the CPU cabinet and how to remove and replace the field replaceable units in the front end cabinet.

NOTE

Some of the modules in the 50 Hz system are different from those in the 60 Hz system. All 50 Hz system differences are noted.

10.3.1 Preliminary Steps

This procedure describes steps that you must take before you begin work on the front end cabinet.

1. On the front of the CPU cabinet, turn the keyed system control panel terminal control switch to the OFF position (Figure 10-1).
2. Open the front and rear doors of the front end cabinet.
3. On the lower left panel at the rear of the front end cabinet, switch the main circuit breaker to the OFF position (Figure 10-3).
4. Contact the customer and ensure that the customer power circuit breaker is switched off.
5. Open the front doors of the CPU cabinet. On the front of the 876-A power controller, switch CB1 to the OFF position (Figure 10-2).

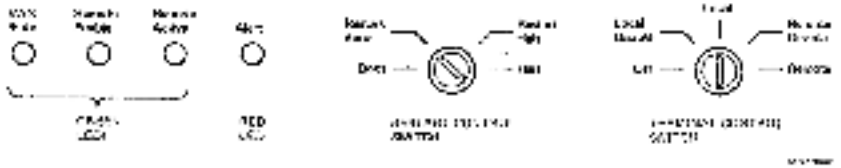


Figure 10-1 System Control Panel

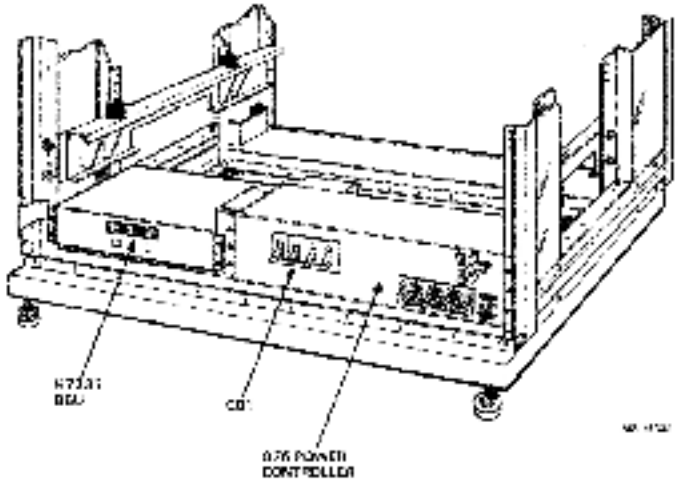


Figure 10-2 876-A Power Controller - Front View

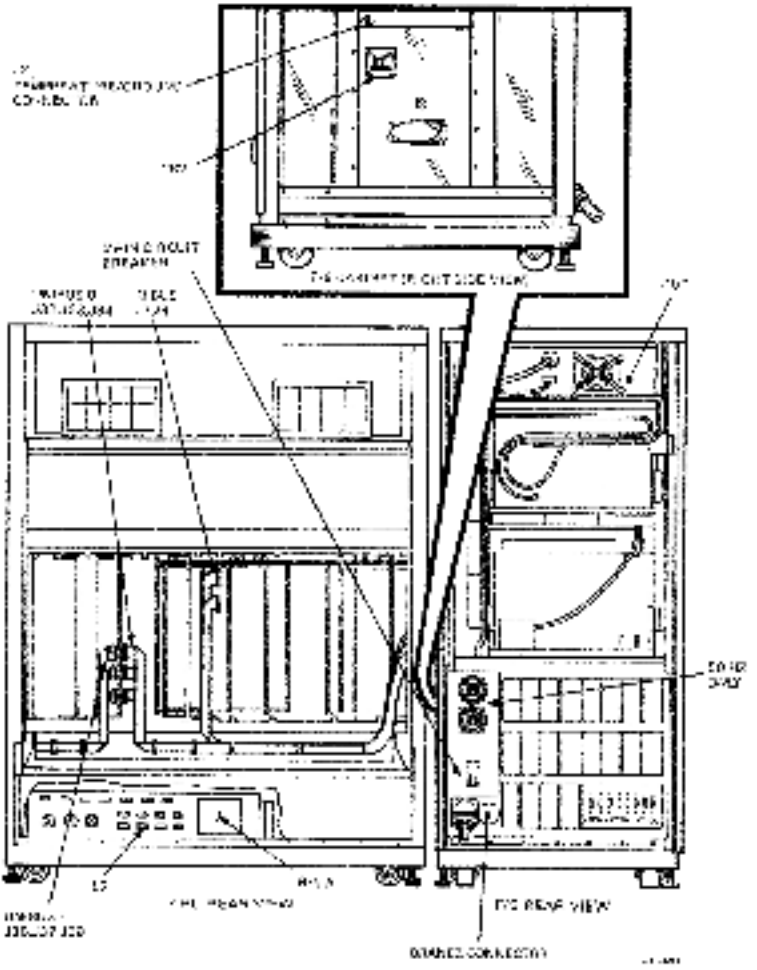


Figure 10-9 Front End Cabinet Cable Connections - Rear View

18.3.2 Front and Cabinet Disconnection

This procedure describes how to electrically disconnect the front end cabinet from the CPU cabinet. Before you begin this procedure, make sure you complete the preliminary steps in Paragraph 18.3.1.

1. Ensure that the customer has turned off power to the system.

2. Disconnect the front and cabinet main ac power cable from the wall receptacle at the customer site.

3. Disconnect triple ribbon cable connectors P1, P5, and P6 from cable connectors J12, J11, and J34 on the I/O backplane of the CPU cabinet (Figure 18-3).

Note

When disconnecting ribbon cables, the notch indicates pin 1 and the red stripe points up.

4. Disconnect double ribbon cable connectors P3 and P4 from cable connectors J7 and J8 on the CPU backplane of the CPU cabinet (Figure 18-3).

5. Disconnect the right angle end of the large ac power cable connector P16 from cable connector J15 at rear of 876-A power controller in the CPU cabinet (Figure 18-3).

Note

When reconnecting the power cable, be careful not to pull out any cable connectors in the CPU cabinet.

6. Disconnect the 8.0Z disk drive ac line cord from receptacle T1 at the rear of the 876-A power controller in the CPU cabinet (Figure 18-3).

7. Disconnect the BA11-A unit ac power cord from receptacle L2 at the rear of 876-A power controller in the CPU cabinet (Figure 18-3).

18.3.3 Constant Voltage Transformer (CVT) Assembly Remove

This procedure describes how to remove the CVT assembly from the front end cabinet. Before performing this procedure, make sure you complete the preliminary steps in Paragraph 18.3.1. Remove the CVT module as follows.

(to be supplied)

18.3.4 RL02 Disk Drive Removal and Installation

This procedure describes how to remove the RL02 disk drive unit from the front end cabinet. Before performing this procedure, make sure you complete the preliminary steps in Paragraph 18.3.2.

18.3.4.1 RL02 Disk Drive Removal - Remove the RL02 disk drive unit as follows.

1. Disconnect the I/O "CABLE IN" connector from the rear disk drive unit (Figure 18-4).

2. Disconnect the ac line cord from the rear of the H7640-A power controller in the CPU cabinet.

3. Disconnect all cable ties that bind the ac line cord to the front end frame.

4. Move to the front of the front end cabinet and carefully slide the RL02 unit out from the front end cabinet until it catches (Figure 18-5).

WARNING

The RL02 disk drive unit is a heavy piece of equipment. The next three steps of this procedure is best handled by two people, one on each side.

5. Remove the two rear slide screws, one on each side of the drive unit (Figure 18-6).

6. Remove the two front slide screws, one on each side of the drive unit (Figure 18-6).

7. Lift the two locking latches on the slide mounting rails on each side of the drive unit (Figure 18-6). Carefully remove the drive.

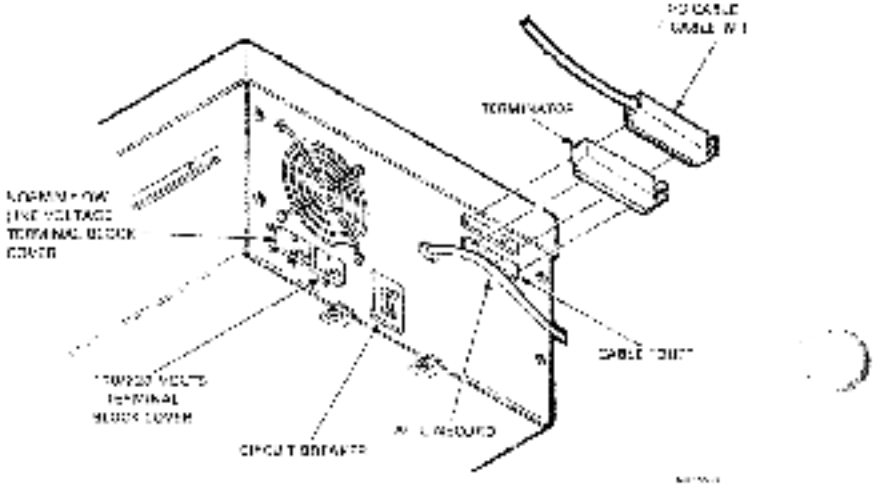


Figure 18-4 RL02 Disk Drive - Rear View

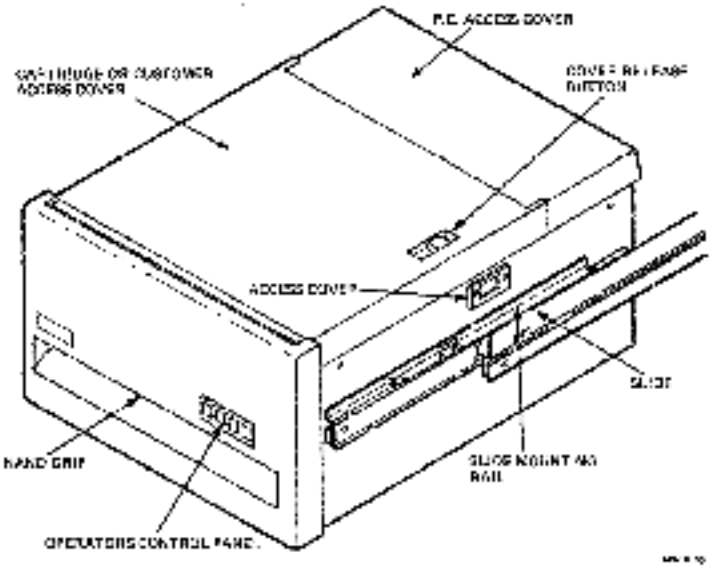


Figure 15-5 RL02 Disk Drive Mounted in Slides

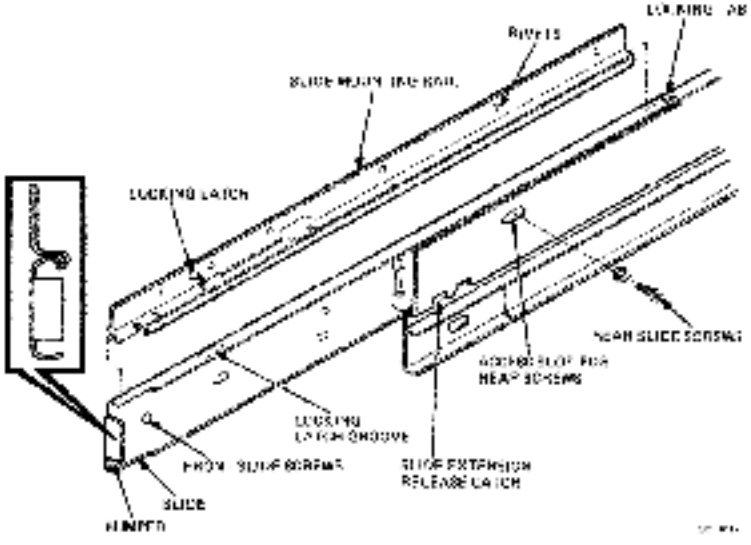


Figure 15-6 Slide Mounting Rail and Slide

10.3.4.2 RL02 Disk Drive Installation - This procedure describes how to install the RL02 disk drive unit into the front end cabinet. Figure 10-5 shows a cabinet-mounted RL02 unit. Install the unit as follows.

1. Open the front and rear doors of the front end cabinet.

2. Remove the slides from the carton containing the disk drive unit. (Retain the hardware for reassembly.)

3. Install the slides in the cabinet using the enclosed hardware. Be sure the slides are at the correct height to permit installation of drawe panels when you finish installing the unit. Also make sure that the slides do not bind on any hardware used to mount the slide.

4. Extend the slides to lock position.

5. Place the drive unit on the slides and reinstall the mounting hardware.

   a. Figure 10-5 shows the relationship between the drive, slide mounting rails, and the slides. Notice the position of the slide mounting rails. These rails are riveted to the sides of the drive.

   b. The cabinet slides fit under the edge of the mounting rails. The forward edge of the mounting rails are curved to grip the curled edge of the slides (Figures 10-5 and 10-6).

   c. At the rear of each slide is a locking tab that grips the top rear edge of the rail (Figure 10-6).

   d. Carefully place the drive on top of the slides, hooking the front and rear of each slide.

   e. Ensure that the locking latch on each mounting rail drops into a groove on each slide (Figure 10-6).

   f. Insert the front slide screws to secure the front of each slide to the drive (Figure 10-6).

   g. Using the slide extension release catch, adjust the length of the slide and insert the rear slide screws. The rear slide screws secure the rear of the slides to the drive.

6. Make sure that the disk drive moves easily on the slides, that there is no binding in the cabinet, and that there is enough room to install the drawe panels.

7. Open the drive access cover by loosening the four captive fasteners holding the cover. Lift the drive access cover off the drive. You can rest the access cover on the rear lip of the drive (Figure 10-7).

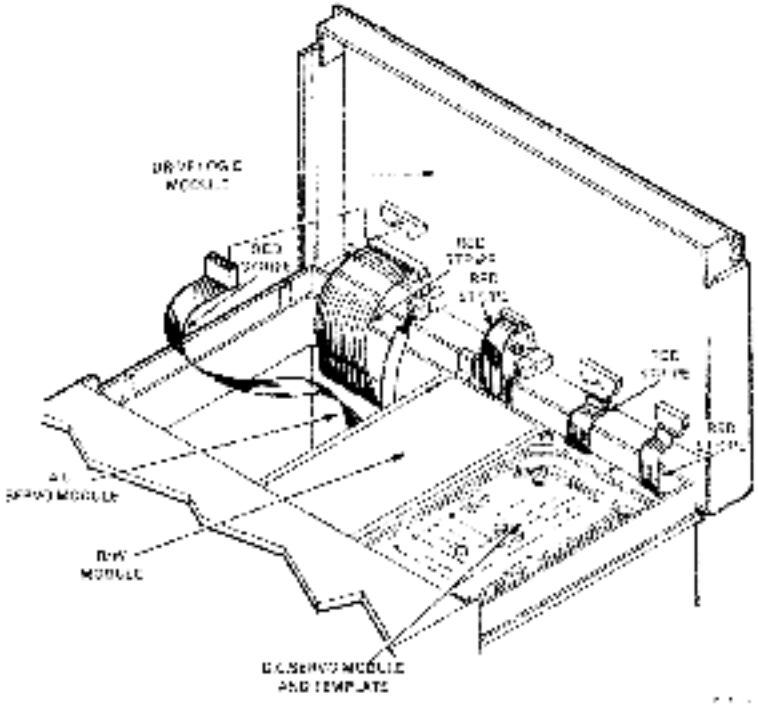


Figure 10-7 8560 Disk Drive with Embossed Drive Logic Module

8. Loosen the head restraining bracket screw on the positioner. Turn the bracket 90 degrees and retighten the screw (Figure 10-8).

9. On the bottom of the drive, remove the two shipping screws that secure the spindle/blower motor.

10. Make sure that the terminal block covers are configured correctly for the input power available (Figure 10-4).

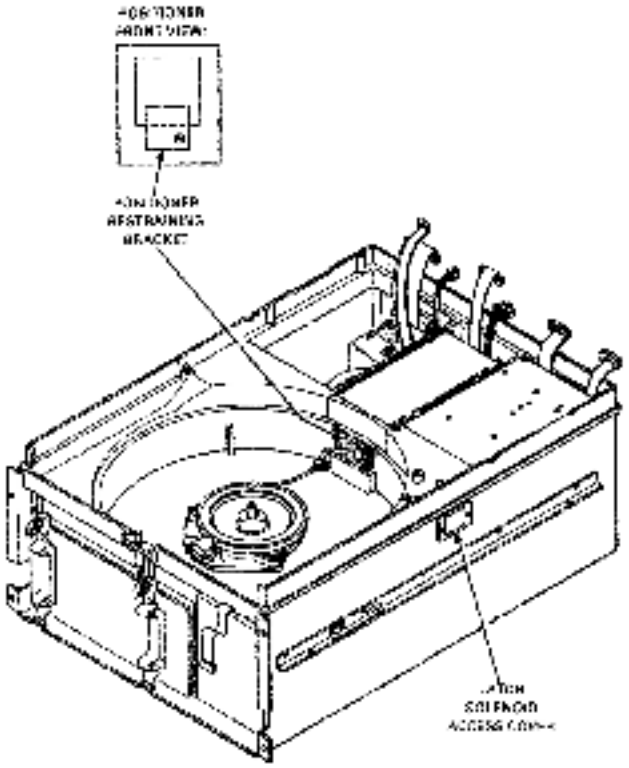


Figure 10-8 RX33 with Covers Removed

11. If there is only one drive in the system, or if this is the last drive of the daisy chain, install a terminator (digital part number 70-12293) on the "CABLE OUT" receptacle at the rear of the drive.

12. If this is a multidrive system, connect an I/O cable from "CABLE IN" of this drive to the "CABLE OUT" of the previous drive. Repeat for each drive.

NOTE

The total length of cable from controller to the last drive must not exceed 30 m (100 ft).

13. Install the proper unit select plug at the front of the drive (Figure 18-9).

14. Connect the ac line cord to receptacle L1 on the rear of the 876-A power controller in the CPU cabinet.

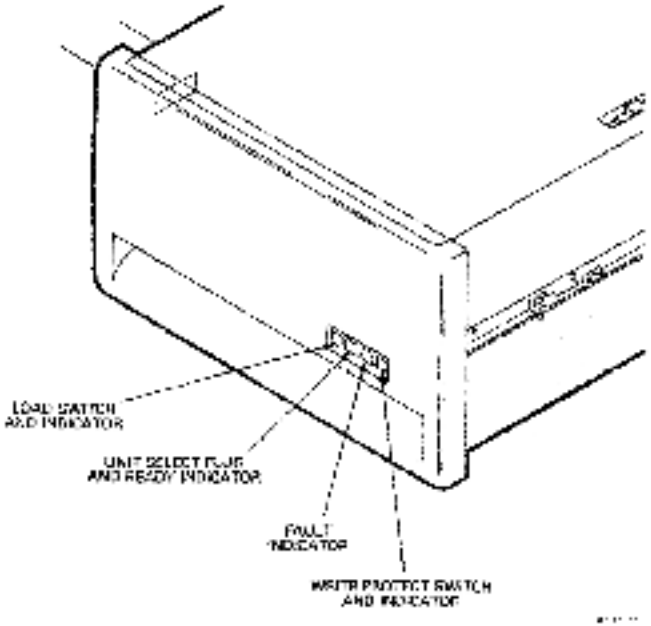


Figure 18-9 RL02 Disk Drive - Front View

18.3.5 BAll-A Unit Assembly Removal and Installation

This procedure describes how to remove and install the BAll-A unit assembly from the front end cabinet. Before performing this procedure, make sure you complete the steps in Paragraph 18.3.1 and 18.3.2.

18.3.5.1 BAll-A Unit Assembly Removal -

1. In the CPU cabinet, disconnect the BAll-A ac power cord plug from receptacle L2 at the rear of the 876-A power controller.

2. Release all fasteners or clamps used to secure the power cord to the frame.

3. At the rear of the BAll-A mounting box, remove the cable clamp bars that secure the external cables to the unit. Each clamp is held by two 1/4 inch nuts (Figure 18-10).

4. Insert a small screwdriver into the slot in the top right of the front bezel. Release the latch that holds the mounting box by sliding the screwdriver to the left (Figure 18-11).

5. Pull the front of the mounting box until it is fully extended and the slide rail levers are engaged (Figure 18-12).

6. Remove and retain the four 6-32 screws that secure the top cover to the mounting box and remove the top cover.

7. Remove the UK7008 cable connector from the backplane, the T/D cables attached to the console interface module (1870901), and all other cable connectors in the modules.

8. Route the cables away from the mounting box.

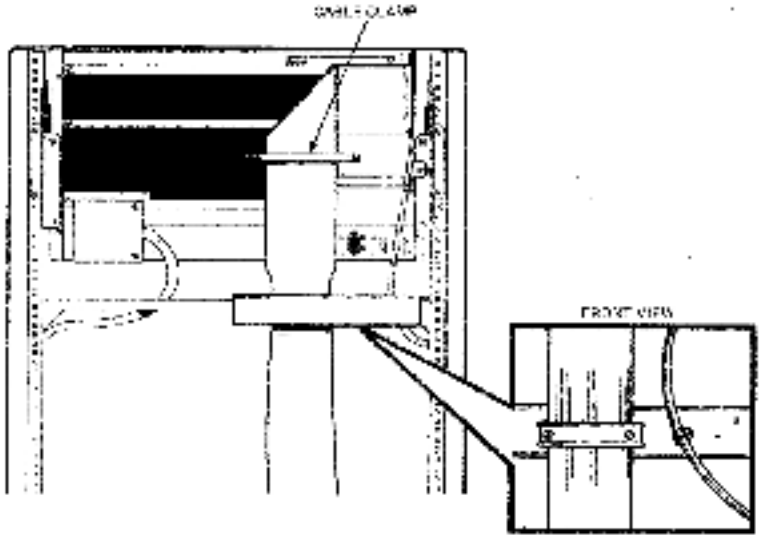


Figure 18-10 Cable Clamps

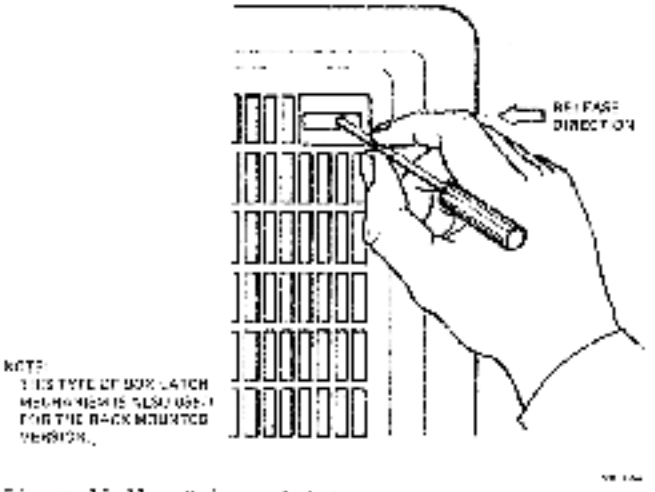


Figure 18-11 Release Latch

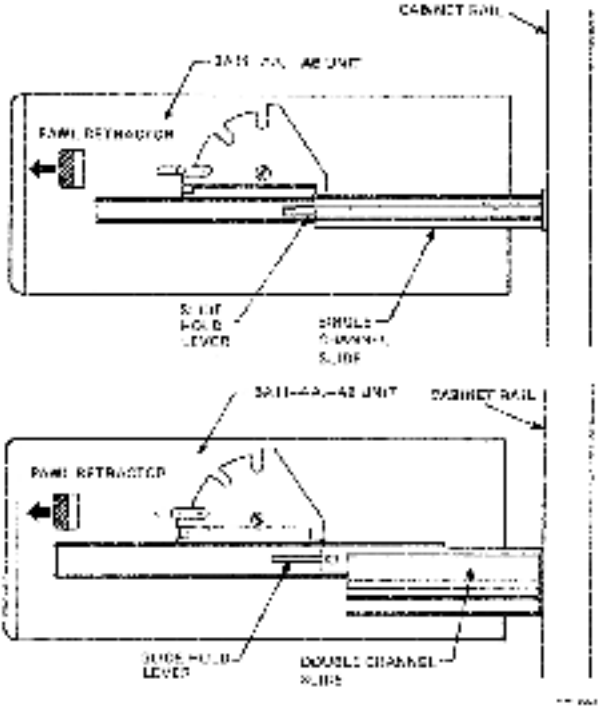


Figure 18-12 RAIL-A Installed In Slide Mount

9. Remove and retain the three 8-32 screws that secure the left and right index plates to the slide assembly (Figure 18-13).

WARNING

The BA11-A mounting box is a heavy piece of equipment. Removal of the mounting box requires two people, one on each side of the unit.

10. Remove the mounting box from the slides by lifting the box until the alignment tabs on the index plates are free from the slide mounting bracket.

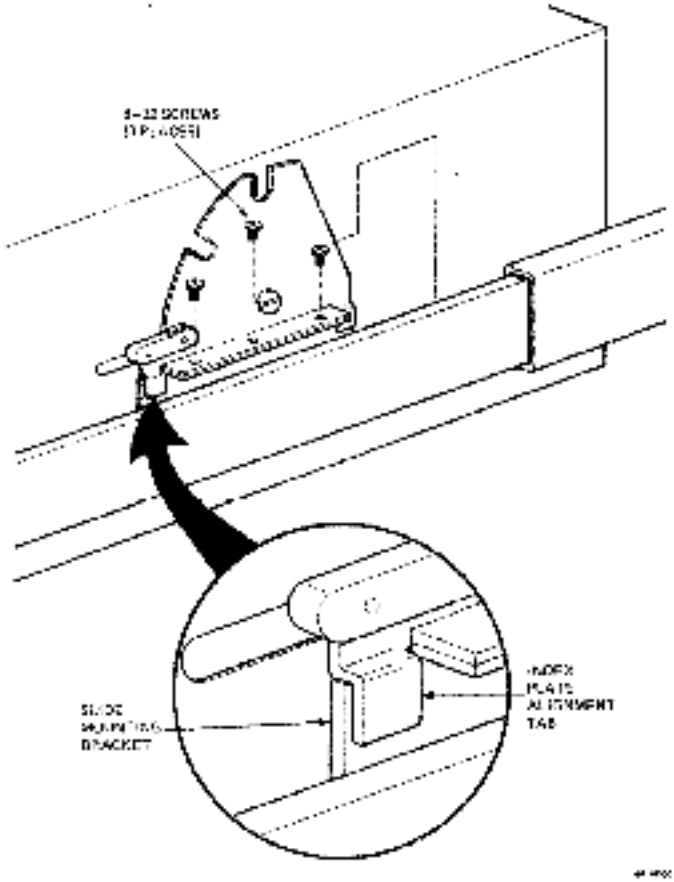


Figure 18-13 Slide to Index Plate Mounting

18.3.5.2 BA11-A Unit Assembly Replacement - This procedure describes how to replace a BA11-A unit assembly in the front end cabinet.

1. Open the front and rear doors of the front end cabinet.

2. Extend the left and right slide channel to their maximum position at the front of the cabinet.

WARNING

The BA11-A mounting box is a heavy piece of equipment. Installation of the mounting box requires two people, one on each side of the unit.

3. Carefully lift the mounting box above the extended slides and set the index plates over the slide mounting brackets on each side of the box (Figures 18-12 and 18-13). The index plate alignment tabs will engage the sides of the slide mounting bracket.

NOTE

When the slides are fully extended, you may have to force the ends of the slides in toward the sides of the mounting box during installation.

4. Insert the three 6-32 screws through the left index plate tab and into the threaded holes of the slide mounting bracket. Tighten the screws (Figure 18-13). Repeat the process for the right index plate.

5. Perform steps 1 through 7 of Paragraph 18.3.5.1, in the reverse order, to complete the replacement process.

18.3.6 H7140 Power Supply Removal

This procedure describes how to remove the H7140 power supply from the BA11-A unit assembly in the front end cabinet. Before performing this procedure, make sure you complete the preliminary steps in Paragraph 18.3.1 and 18.3.2.

1. In the CPU cabinet, remove the BA11-A ac power cord from receptacle L2 at the rear of the H76-A power controller.

2. Release all fasteners or clamps used to secure the ac power cord to the cabinet frame.

3. At the rear of the BA11-A box, remove the cable clamp bars that secure the external cables to the box. Each clamp is held by two 1/4 inch nuts (Figure 18-10).

4. Remove and retain the four 6-32 screws that secure the top cover to the mounting box, and remove the top cover.

5. Remove the I/O bus cables from the cable trough at the left side of the power supply and route the cables over the top of the power supply.

6. Remove and retain the two 8-32 screws located in each of the two chassis angles at the rear of the mounting box (Figure 18-14).

7. Release the panel retractors on each side of the box and tilt the box 90 degrees to the maintenance position (Figure 18-15).

8. Remove and retain the four 6-32 screws that secure the bottom cover to the mounting box (Figure 18-13). Remove the cover.

9. Remove and retain the four 6-32 screws that secure the cover plate to the bottom of the power supply assembly (Figure 18-15). Remove the cover.

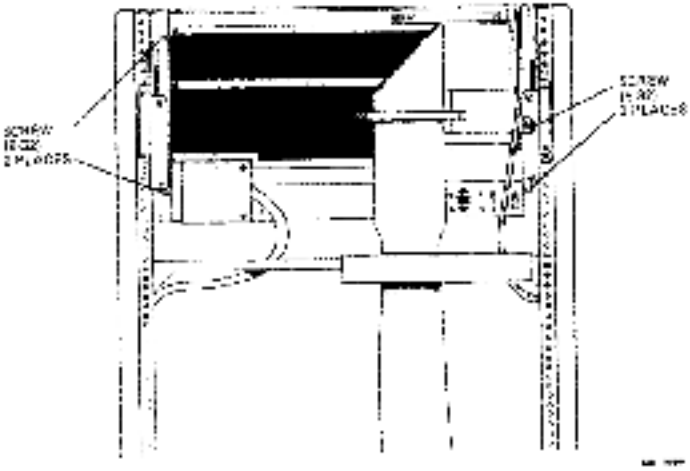


Figure 18-14 Power Supply Unit, Rear Mounting Screws

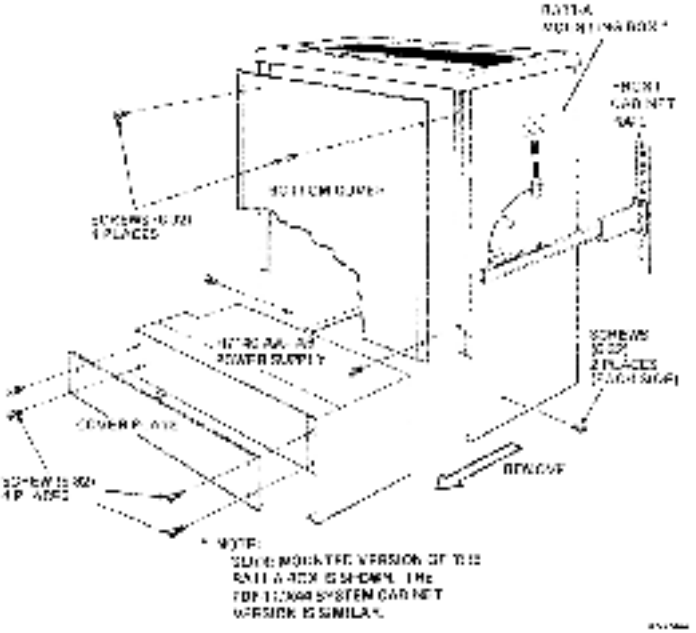


Figure 18-15 Power Supply Unit Removal

10. Remove and retain the 10-32 screw that secures the ground lead to the ground bus (Figure 10-16).

11. Loosen the two 3/8 inch nuts on the clamp holding the flexprint cable to the ground bus bar.

12. Loosen the two 3/8 inch nuts on the clamp holding the +5 V flexprint cable to the +5 V bus bar.

13. Slide the ground and +5 V cables away from the clamps and bend the cables up toward the backplane.

14. Remove the power flexprint connector P1 from the power supply connector J11 and bend the connector up toward the backplane.

15. If one or more additional backplanes are mounted in the box, disconnect the connectors attached to J2, J3, and J4 of the power distribution board. Disconnect the backplane connectors from P2, P3, and P4 of the power distribution harness.

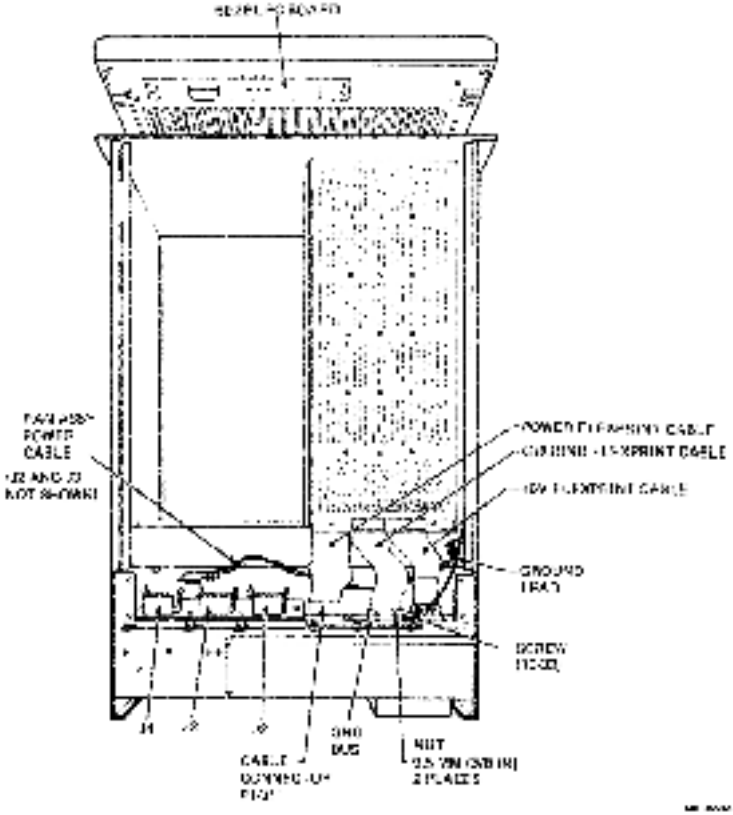


Figure 10-16 Mounting Box, Power Cable Connection

16. Remove and retain the 6-32 screws located on each side of the mounting box, toward the rear (Figure 18-15).

17. Slide the power supply assembly forward about 5.0 cm (2.0 inches) and disconnect the fan assembly power cable from connectors J2 and J3 on the power supply PC board.

18. Slide the power supply assembly from the mounting box and away from the cabinet.

NOTE

When replacing the power supply, make sure you set the mounting box in the maintenance position. Then perform the reverse of steps 16 through 1 above.

18.3.7 Fan Panel Assembly Removal (50 Hz system)

This procedure describes how to remove the fan panel assembly from the front end cabinet and how to remove each fan from the panel. Before performing this procedure, make sure you complete the preliminary steps in Paragraph 18.3.1 and 18.3.2.

NOTE

This procedure is for 50 Hz systems only. The 60 Hz system does not have a Fan Panel Assembly.

1. Remove the ten screws surrounding the fan panel then carefully pull out the entire panel until it catches on the fan power cable (Figure 18-5).

2. Pull the power cable connector away from the AC plug on the power supply (Figure 18-17).

3. Remove each fan by removing the four screws surrounding the fan.

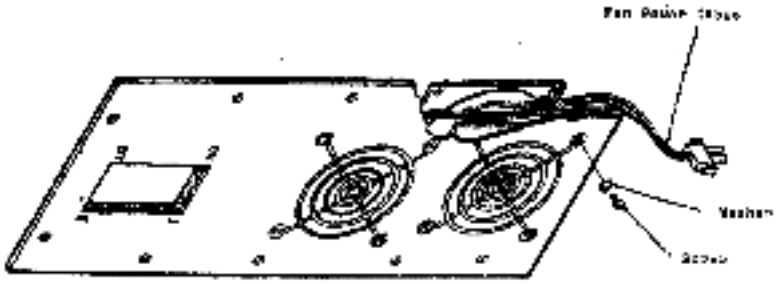


Figure 18-17 Fan Panel Assembly

10.3.8 Front door filter removal

This procedure describes how to remove the two filters from the front door of the front and cabinet. Before performing this procedure, make sure you complete the preliminary steps in Paragraph 10.1.1.

The filters are on the inside of the door. Remove each filter by peeling it away from the velcro strip holding it in place (Figure 10-16). The part number for the filters are: top, 12-11265-06; and bottom, 12-11265-14.

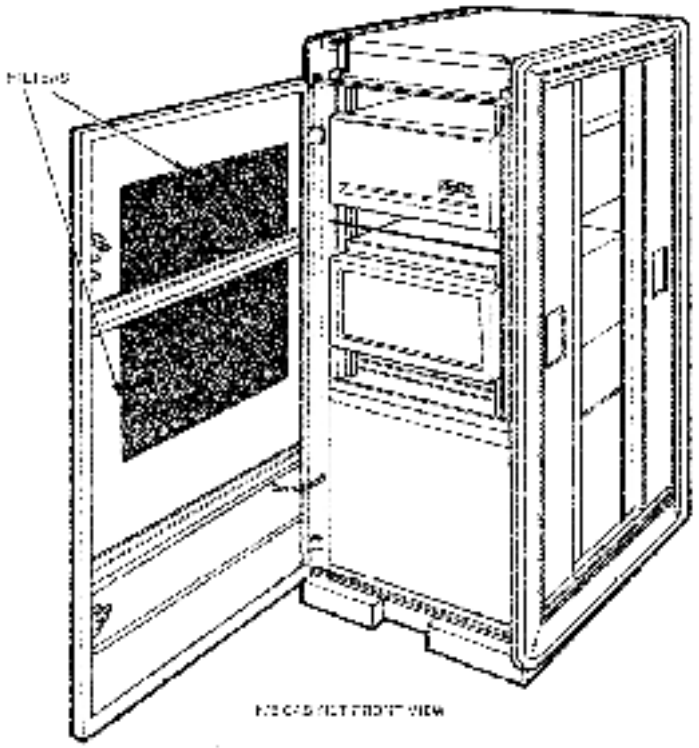


Figure 10-16 Front End Cabinet - Front Door Filters

18.3.9 Rear Door Filter Removal

This procedure describes how to remove the three filters from the rear door of the front end cabinet. Before performing this procedure, make sure you complete the preliminary steps in Paragraph 18.3.1.

The filters are on the inside of the door. Remove each filter by peeling it away from the velcro strip holding it in place. The part numbers for the filters are: top, 12-11255-02; and middle and bottom, 17-11255-00. (Figure 18-19).

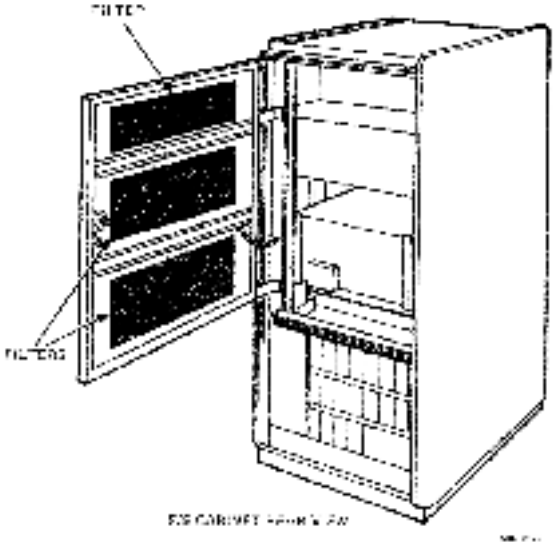


Figure 18-19 Front End Cabinet - Rear Door Filters

18.4 CPU CABINET ASSEMBLY

This section describes how to remove and replace the major field replaceable units in the CPU cabinet assembly.

18.4.1 Preliminary Steps

Complete the following steps before you do any work on the CPU cabinet.

1. On the front of the CPU cabinet, turn the keyed system control panel terminal control switch to the OFF position (Figure 18-20).

2. Open the front and rear doors of the CPU and front end cabinets.

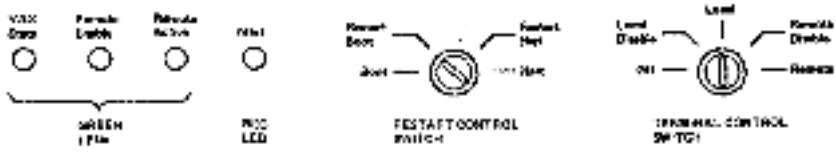


Figure 18-20 System Control Panel

16.4.2 Centrifugal 1500 CFM Blower Removal

This procedure describes how to remove the Centrifugal 1500 CFM Blower from the CPU cabinet. Before performing this procedure, make sure you complete the preliminary steps in Paragraph 16.1.1.

1. On the lower left panel at the rear of the front and cabinet, turn the main circuit breaker to the OFF position.

2. On the front of the 816-A power controller in the CPU cabinet, switch CB1 to the OFF position (Figure 16-2).

3. Remove the two screws from the rear of the top cover of the CPU cabinet (Figure 16-2).

4. Lift off the top cover.

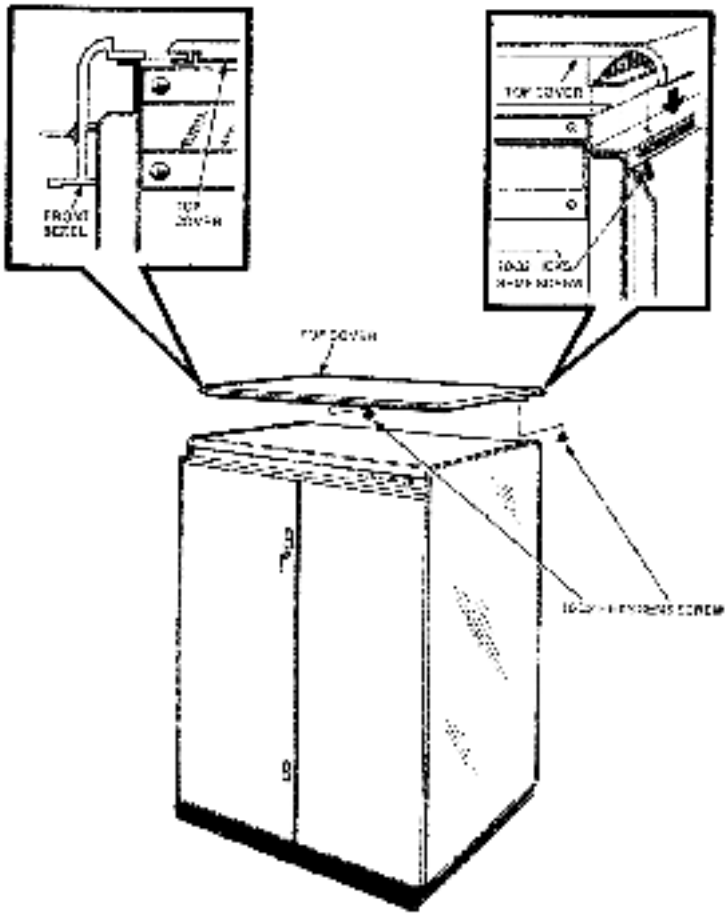


Figure 16-21 CPU Cabinet - Top Cover Removal

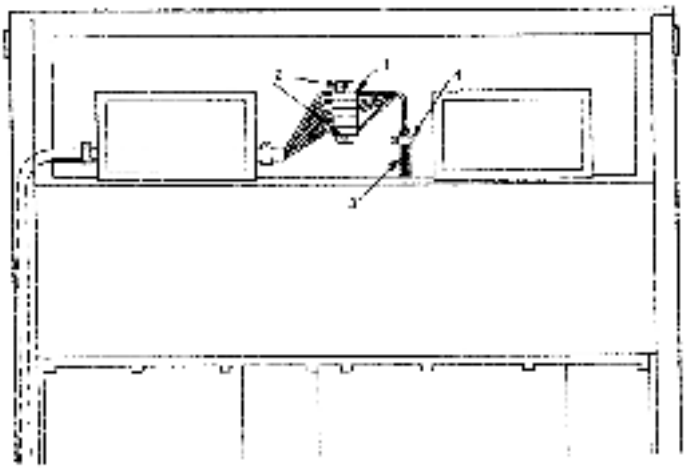
5. At the rear of the CPU cabinet, disconnect the blower leads from terminal block 1 (Figure 18-7A).

6. Loosen the blower cable clamp and release the blower cable (Figure 18-22).

7. On top of the CPU cabinet, remove the 2S screws securing the blower to the CPU cabinet frame (Figure 18-23).

WARNING

The centrifugal blower is a heavy piece of equipment. Removal of the blower should be done by two people.



CPU CABINET - REAR VIEW

NOTES:
1 TERMINAL BLOCK
2 SCREWS (2)
3 BLOWER
4 CABLE CLAMP

Figure 18-22 CPU Blower Cable Removal

8. Using the handles on top of the blower, lift the blower up and away from the CRT cabinet frame.

NOTE

Upon power up after installation of a new blower assembly, ensure that the blower fans rotate in the downward direction (as viewed from the rear of the cabinet, looking into the blower exhaust port). If the direction of rotation is reversed, check the harness connection on the terminal block (Figure 10-22) for reversed leads.

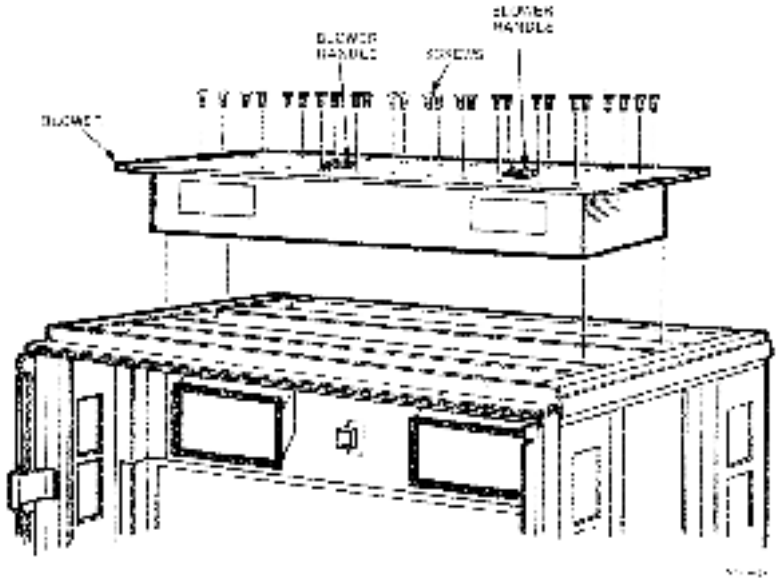


Figure 10-23 CRT Blower (Removal) - Rear View

18.4.7 Modular Power Supply Removal

This procedure describes how to remove individual modular power supplies from the power supply assembly in the CPU cabinet. Before performing this procedure, make sure you complete the preliminary steps in paragraph 18.4.1.

1. On the front of the RTA-4 power controller, switch CB1 to the OFF position (Figure 18-2).
2. Select the modular power supply you want to remove from the power supply assembly. Loosen the thumbscrew at the bottom of the module.
3. Using the handle at the middle of the module, slide the module out of the Power Supply Assembly (Figure 18-24).

NOTE

You can remove all of the modular power supplies from the Power Supply Assembly using the above procedure.

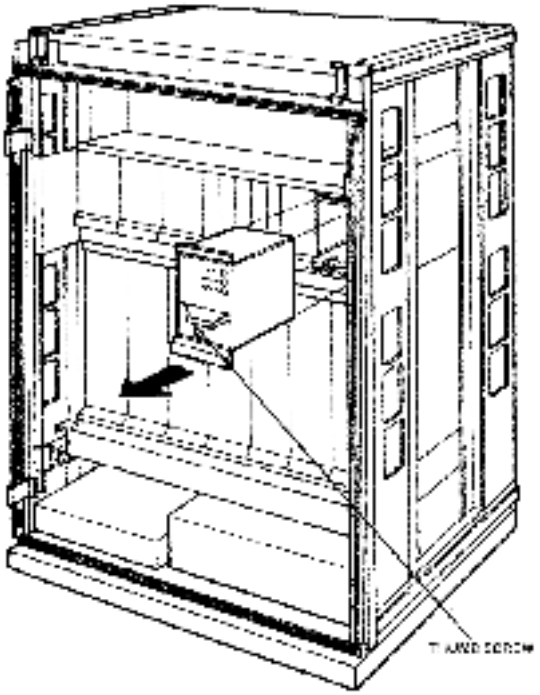


Figure 18-24 Modular Power Supply Removal

4. To replace the modular power supply, simply repeat steps 1 - 3 above in the reverse order. Exercise caution when the regulator is being "mated" to the backplane and bus bar assemblies.

18.4.4 CPU Module Replacement

This procedure describes how to remove module boards from the card cage assembly. Before performing this procedure, make sure you complete the preliminary steps in Paragraph 18.4.1. The cleaning procedure for module paddle fingers is found in Section 18.5.

1. If replacing the DC29I console module, turn the PWR ON/OFF switch to the OFF position. It is located on the H7231 battery backup unit (Figure 18-21).

2. Connect the grounding cord, on the side of the CPU cabinet frame, to the plug on the module case (Figure 18-25).

3. Strap the velcro wrist strap, connected to the cabinet frame, around your wrist.

4. Break the recessed seal on the left side of the module case and open the case.

5. Place the module case on the floor so that the foam padding on the upper half of the case is exposed.

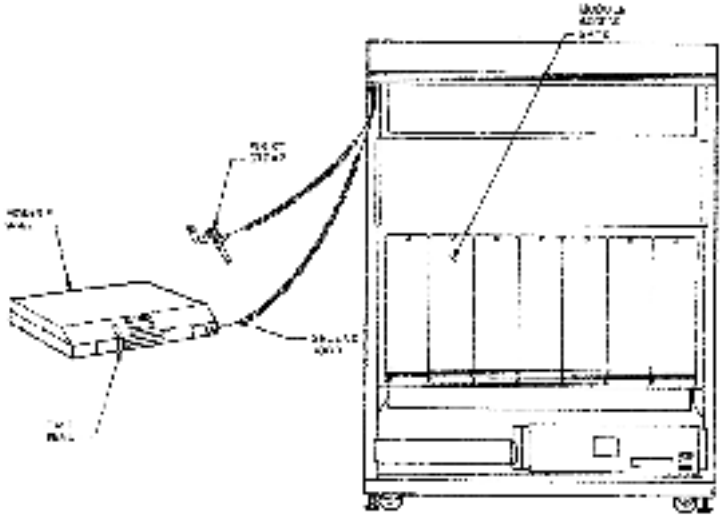


Figure 18-25 Module Case Ground Connection

6. Choose the module board you want to replace. Turn the wing nut at the top of the hinged gate covering that module board and open the gate (Figure 18-26).

7. Grasp the bad module board inside the card cage and carefully work it out.

8. Place the bad board on the foam padding on the upper half of the module case. Take the good module board out of the module case and place it into the empty slot.

9. Close the hinged gate and latch the wing nut at the top of the gate, and, if the IOV power was turned off at the 8723, turn it on.

10. Secure the bad board in the module case.

11. Disconnect the grounding cord from the module case, then remove the strap from your wrist.

12. On the Reason for Return label on the lower right side of the module case, make sure you indicate what is wrong with the bad module, your name, and badge number in the spaces provided.

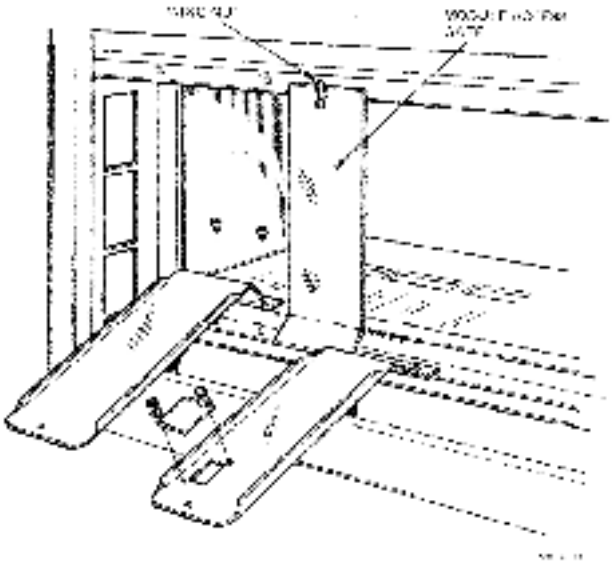


Figure 18-26 CPU Module Access Gate

18.4.5 Array Modules

The 64 Kb, 16 Mb, or 4 Mb array modules are removed as previously described for CPU modules. However, if a 64 Mb or 16 Mb array module is to be replaced, the DMB (standard memory until) will have to be removed and reinstalled on the new array module. To remove an SMU from the 64 Mb or 16 Mb array, use the following procedure:

1. Remove the two mounting screws from the faulty SMU, or disengage the SMU from the four standoff fingers.

2. Carefully work the SMU free of its connector on the native board.

3. Reverse the previous two steps to install a replacement SMU, ensuring that the standoffs are locked, and then install the memory module.

18.4.6 Air Filter Removal

This procedure describes how to remove the air filter from the card cage support assembly.

MARKING

> Before performing this procedure, make sure you have completed the preliminary steps in Paragraph 18.4.1. The CPU must be powered down. If not, the CPU will power down automatically if you try to remove the air filter. You cannot power up the CPU without an air filter in the proper position.

1. Turn the two black catches on the bottom of the card cage support assembly so that they are parallel to the support assembly (Figure 18-27).

2. Slide the air filter out of the CPU cabinet.

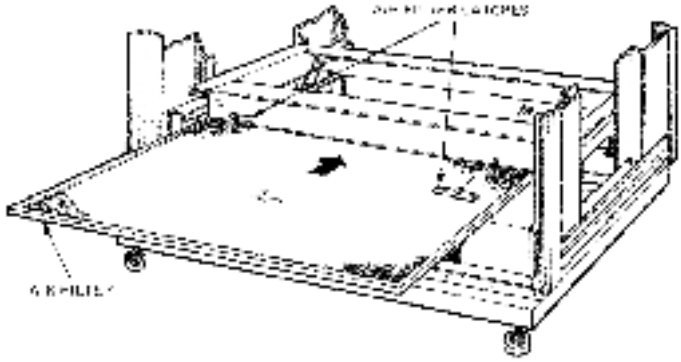


Figure 18-27 CPU Card Cage Air Filter Removal

18.4.7 Air Flow Sensor Removal and Installation

This procedure describes how to remove the air flow sensors from the card cage assembly. Before performing this procedure, make sure you complete the preliminary steps in Paragraph 18.4.1.

18.4.7.1 Air Flow Sensor Removal -

1. There are two air flow sensors on the card cage support assembly (Figure 18-22). Disconnect the cable connector from the air flow sensor (P1 or P3).

2. Remove the screw from the middle of the air flow sensor and remove the air flow sensor.

18.4.7.2 Air Flow Sensor Installation -

1. Installation is the reverse of the removal procedure.

2. Test the new sensor after powerup by giving a SHOW POWER command. The sensor is good if no fault is reported.

18.4.8 Air Temperature Sensor Replacement

This procedure describes how to remove the temperature sensors from the CPU cabinet. Before performing this procedure, make sure you complete the preliminary steps in Paragraph 18.4.1.

18.4.8.1 Temperature Sensor Removal -

1. There are four temperature sensors; three on the card cage assembly, and one on the card cage support assembly. Disconnect the BMM harness cable spade lugs from the temperature sensor you are removing (Figure 18-20).

2. Remove the screw from the right side of the heat sensor and remove the temperature sensor.

NOTE

For troubleshooting or emergency situations only, the system may be operated with one or more temperature sensors removed or disconnected. Once a sensor is disconnected, the console (if in PIO mode) will report the condition via a message as follows:

?BMM DETECTED TEMPERATURE INPUT YK IS OPEN OR VERY COLD. THIS IS A WARNING AND WILL NOT INITIATE A POWER DOWN SEQUENCE.

18.4.8.2 Temperature Sensor Installation - The temperature sensor installation procedure is the reverse of the removal procedure.

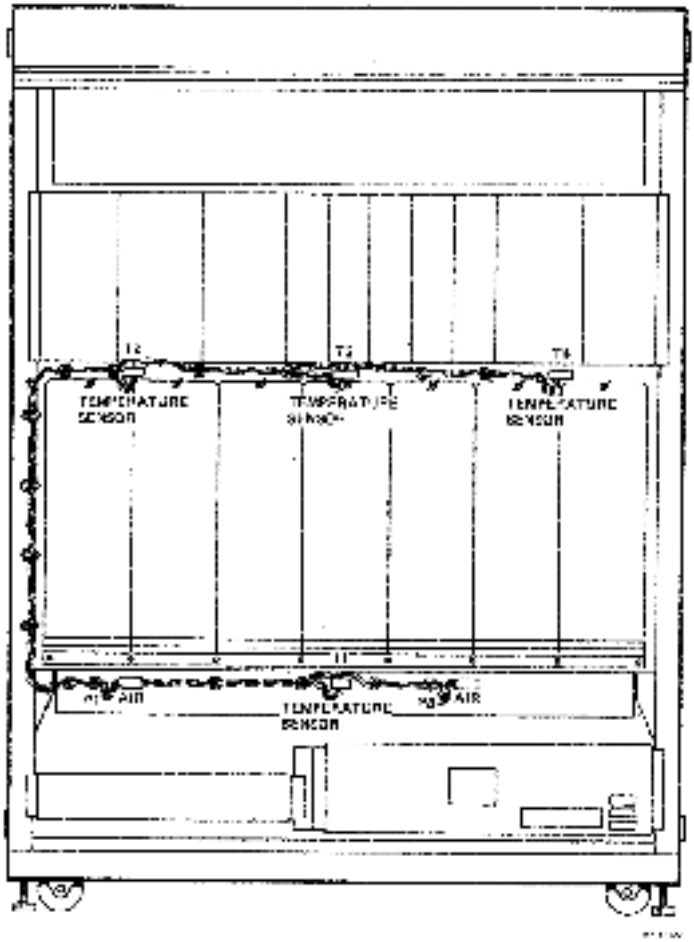


Figure 10-26 Temperature and Air Sensor Locations

18.4.9 876-A Power Controller Removal

This procedure describes how to remove the U76-A power controller from the CPU cabinet. Before performing this procedure, make sure you complete the preliminary steps in paragraph 18.4.1.

1. Disconnect the ac distribution cable from J16 at the rear of the 876-A (power to the U76-A).

2. On the lower left panel at the rear of the front and cabinet, switch the main circuit breaker to the off position. This action cuts off power to the power controller.

3. Disconnect MPS cable connectors from outlets J11 and J12 at the rear of the power controller (Figure 18-29).

4. Disconnect the blower assembly cable connector from outlet J13 at the rear of the power controller.

5. Disconnect BBC "BBC Out bus" cable connector from outlet J6 at the rear of the power controller.

6. Disconnect BBC "Relay Out" cable connector from outlet J7 at the rear of the power controller.

7. Disconnect BBU power connector from J17 at the rear of the power controller.

8. Disconnect MPS "Total Off" connector from outlet J8 at the rear of the power controller.

9. Disconnect MPS "Alarm" 6 wire/white cable connector from outlet J10 at the rear of the power controller.

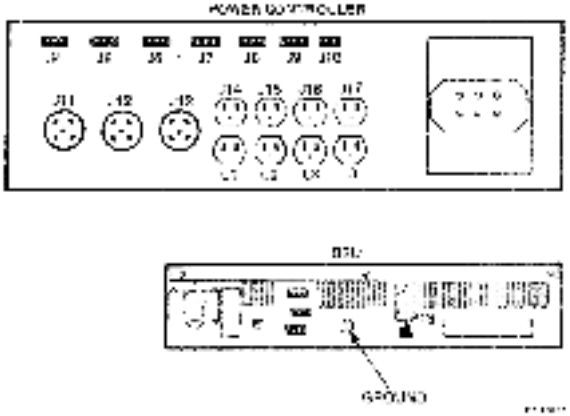


Figure 18-29 Power Controller and BBU - Rear View

10. Disconnect the 8inch disk drive and BAll-A unit assembly power cords from outlets L1 and L2 (respectively) at the rear of the power controller.

11. Move around to the front of the CPU cabinet and remove the six screws, three on the left and three on the right side of the power controller (Figure 18-30).

WARNING

The 870-A power controller is a heavy piece of equipment. Be careful when sliding out the old power controller and inserting the new. The task of removal and insertion is best handled by two people.

12. Slide the power controller out from the CPU cabinet.

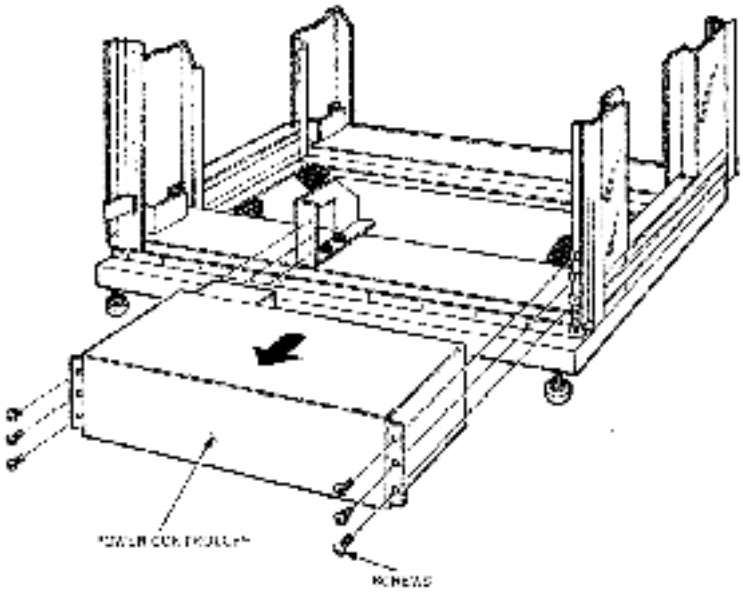


Figure 18-30 Power Controller Removal

10.4.10 B7231 BBU Power Supply Removal

This procedure describes how to remove the B7231 battery back-up power supply from the CPU cabinet. Before performing this procedure, make sure you complete the preliminary steps in paragraph 10.4.1.

1. On the front of the 876-A power controller, switch CB1 to off to remove power to the BBU. Then turn the BBU ON/OFF switch on the front of the BBU to the off position (Figure 10-2).

2. Disconnect cable connector P10 from the rear of the BBU power supply (Figure 10-31).

3. Disconnect bus bar ground connector and bus bar cable connector from the rear of the BBU power supply.

4. Disconnect BBU main power cord connector from outlet J3 at the rear of the 876-A power controller.

5. Disconnect black/white cable connector P6 from the rear of the BBU power supply.

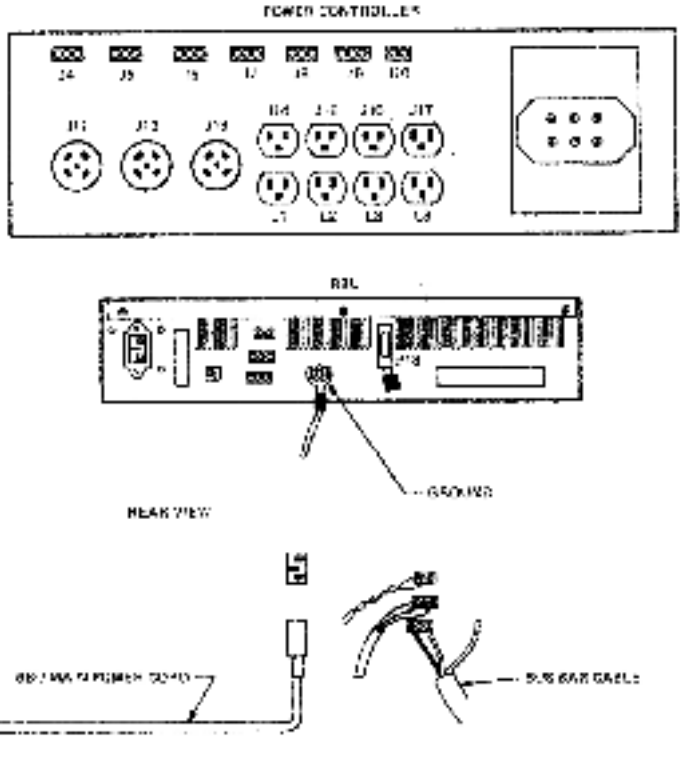


Figure 10-31 BBU Cabling

6. Move around to the front of the CPU cabinet and remove the two securing nuts on the sides of the power supply (Figure 10-32).

7. Slide the power supply forward out from the CPU cabinet.

8. Remove the four screws, two on the left and two on the right side, from the power supply shell.

9. Slide the power supply shell out from the CPU cabinet.

10. A new H7231 power supply comes with shell sleeves. Reverse the above procedure to install the new power supply.

NOTE

Make sure the voltage select switch on the front of the power supply is set correctly for your environment (120 V). Also, make sure that the TOY ON/OFF switch is set to the on position when the system is powered-up.

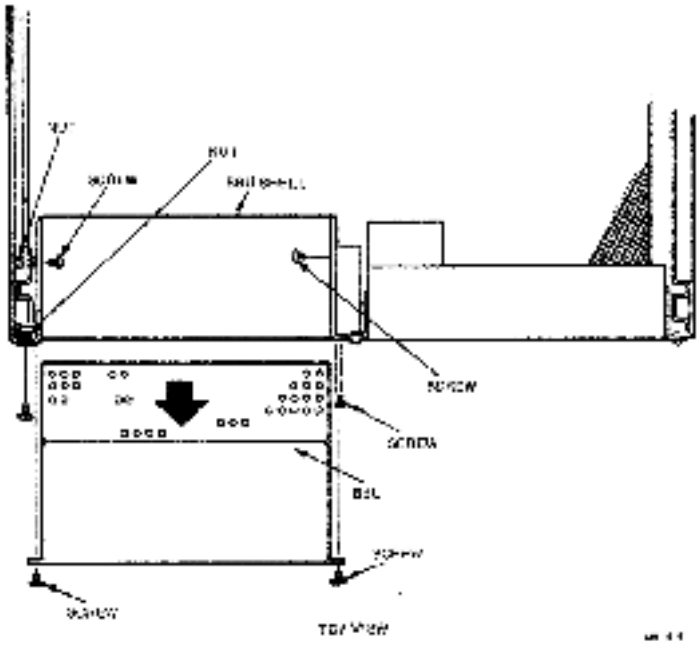


Figure 10-32 DHB Removal

18.5 MODULE PADDLE CONNECTOR CLEANING PROCEDURE

18.5.1 Introduction

This procedure describes the cleaning method for the gold plated module paddle connector fingers.

NOTE

This procedure replaces the traditional eraser method - which removes up to 10 microinches of gold each time an eraser is used for cleaning.

The continuous gold removal causes corrosion of the base metals (copper and nickel) under the gold, which in turn causes contact intermittents and opens.

For additional detail refer to:

1. Engineering Standard EL-F8266-00, Title: Printed Wiring Board (PWB) Gold Contact Cleaning Process

   DOCUMENT IDENTIFIER: ES-EL-F8266-00

2. Miscellaneous Field (Green) Field Service Miscellaneous Tech Tip Number 1, April 6, 1984, Title: Gold Finger Cleaning Using Gold Wipes

18.5.2 Required Materials

1. Cleaning material: EXPON GOLD WIPES Cleansing Pads, Digital part number: 49-01603-01.

2. Protective gloves, SOLVEX gloves by Edmont, vendor part number 37-175-X (where X-= size). There is no Digital part number at this time.

18.5.3 Precautions

1. DO NOT USE gold wipes to clean any part of a disk or tape assembly such as disk heads and packs, tape heads and magnetic tapes, etc.

2. NO SMOKING - Highly toxic phosgene gas can develop from fibers coming into contact with a burning cigarette tip.

3. Do not expose Gold Wipe pads to sources of extreme heat (e.g., soldering iron) as toxic gases can be generated.

4. Use with adequate ventilation.

5. Avoid prolonged contact with skin and eyes.

6. Wear the specified gloves. Do not use latex or polyethylene gloves - they will dissolve and leave harmful plasticized film on the contact surfaces.

7. Wear safety glasses with side shields.

8. Contact lenses should not be worn.

9. After using Gold Wipes, wash hands thoroughly with soap and water before handling food or drink.

10. Do not use Gold Wipes on rubber, polystyrene, aluminum, and most plastics.

18.5.4 Cleaning Procedure

1. Ground yourself properly using a grounding strap.

2. Follow the previous PRECAUTIONS.

3. Place the module on a static mat, or on a clean, conductive, and grounded table surface.

4. Use one Gold Wipe cleaning pad for each module.

5. Open a RX809 packet, and remove the cleaning pad.

6. Firmly grasp the module to be cleaned by the handle, or an unpopulated area of the module - minimizing the risk of static and physical damage.

7. Hold the pre-folded pad partially open.

8. Place the module paddle fingers into the partially opened pad and apply finger pressure against the pad.

9. Maintain finger pressure while pulling the pad away from the paddle fingers. As shown in Figure 18-33, the cleaning action should be parallel to the length of the fingers.

10. Wipe all paddle fingers covering the foil of the pad as required.

11. After cleaning each module, discard the pad and packet in a disposal receptacle which is located in an adequately ventilated area.



Figure 18-33 Paddle Finger Cleaning Action

CHAPTER 19

TECHNOLOGY AND TOOLS

19.1 MCA DESCRIPTIONS, LOCATIONS, AND PRINT SET CROSS REFERENCE

Table 19-1 contains an alphabetical list of MCAs, with their mnemonics, name descriptions, and print set locations.

Table 19-1 MCA/Print Set Cross Reference

MCA	Description	Print Location (Quantity)
ABB	ABus Interface	MCC4
ADA	Address translate	MAP1(3)
		MAP2(2)
ADD	Address translate	MAP2
ADD	CPRA/CPSB address, WBus match A/B, RBus WBus address	EDPC
ALU	BVA, ARbus, BROT	EDP1(2)
		EDP2(2)
		EDP3(2)
		EDP4(2)
ARB	Arbitor	MCC6
ACB	Adder Control B	FA13
ACC	Adder Control C	FA13
ACL	Adder Carry Look-ahead	FA10
ALN	Alignment	FA04(2)
		FA05(2)
BSA	UPC, micro stack address counter	CSBP(3)
CAM	Match	MAP3(2)
CKP	Check bit	MCDM
CLA	Carry Look-ahead	EDP9
CRA	Control Register A	MCC7
CRB	Control Register B	MCC5
DBS	Double Buffer Slice	ISD6(2)
		ISD7(2)
DPC	Data Patch Control = ALU control, WBus match control, parity control	EDPD
DPP	Data Path Parity and carry look-ahead	IDP6(2)
ECC	ECC	MCDM
ERR	Error	MCCM

Table 19-1 MCA/Print Set Cross Reference (Cont.)

MCA	Description	Print Location (Quantity)
FBS	FBox Registers	FA01
FAU	Fraction ADDER	FAU6(2)
		FA07(2)
		FA08(2)
		FA09(2)
FXP	F Exponent Processor	FA01
GXP	G Exponent Processor	FA01
IAD	IBox Adder	IDP1(4)
		IDP2(4)
IDT	Instruction Buffer	IBD1(5)
		IBD2(3)
		IBD3(3)
IBC	Decode Flags, Buffer control	IBUA
IOP	OP Bus data path	IBD4(2)
		IBD5(2)
IST	IBox Stall	ICNG
IVA	IBox Virtual Address	IDP3(4)
		IDP4(4)
MAI	MBox Array Interface	MCC8
MAX	Multiplier Accumulate Extension	FM11
MCF	Microcontrol Field Decode	MUX6
MCB	Multiply Control B	FM09(2)
		FM10(2)
MCL	Multiply Carry Look-ahead	FM08
MDP	Memory Data Paths	MCD1(2)
		MCD2(2)
		MCD3(2)
MIC	UPC, CHAR load sync., clear flag sync.	CBBQ(2)
MMD	MBox Microcode Data Registers	MCCA(2)
		MCCB(2)
MMS	MBox Microsequencer	MCC1
MPA	Multiplier Unpacker	FM01(2)
		FM02(2)
MPY	Multiplier	FM03(2)
		FM04(2)
		FM05(2)
		FM06(2)
		FM07(1)
MPS	Multiplier Extension	FM07
MSQ	Micro Sequencer (FBox model	FA11
MSQ	Micro Sequencer (FBox multiplier)	FM12
PDP	Data path parity checker - CPBA/B, ALU, MBUS, ALU shift bits, VMQ shift bits OPBus long word, MReq longword	FUPU
REG	Address Registers	MAPP
RES	Response	MCC5
RMT	RAM Register Control	FA12

Table 19-1 MCA/Print Set Cross Reference (Cont.)

MCA	Description	Print Location (Quantity)
SCR	Shift Count Element - shift count information and control, ALU control, data path control, status receivers	EDPE
SKU	16bit Micro Sequencer	TOS6
SHF	Shifter - Dual ALU shifting, data formatting, 0 - 63 data rotation, MBus receiver	EDPA[2] EDPB[3] [AU2]<2]
SOP	Source Operand	[BUF]
SPA	Scratch Pad Addressing	IBUF
SPM	Specialist Decode	IBUF
STA	Port Status	VCC7
TRO	Data Register	MCDQ
TPC	17bit Micro PC	ICA8
UPK	Unpacker	IOP6
VAL	IBuf Valid Bits	IBUF
WVP	Write Valid	MADI

Table 19-2 contains a list of MCAs by module, with corresponding print set locations.

Table 19-2 MCA Descriptions and Locations

Module Name	MCA Name	Qty	Location	Description
ESda MCAs				
L0209 (EDP) EBox Data Path				
	ADD	1	C	Address
	ALU	3	A-1	Arithmetic Logic Unit
	CCR	1	J	Condition Codes
	CLA	1	K	Carry Lock Ahead
	DPC	1	D	Data Path Control
	PDP	1	H	Parity Data Path
	SCE	1	L	Shift Counter
	SHF	4	A-B	Shifter
L0210 (CSA) Control Store B				
	PPA	5	P	Branch Enable
	MIC	2	Q	Microsequencer

Table 19-2 MCA Descriptions and Locations (Cont.)

Module MCA	MCA Name	Qty	Location	Description
EBOX MCAs				
L0212 (FEA, FA) FBox Adder				
	ACB	1	14	Adder Control B
	ACC	1	15	Adder Control C
	ACL	1	10	ACC Carry Look Ahead
	ALN	4	4-5	Aligner
	EAD	8	6-9	Adder
	FBR	1	1	Register
	FAS	1	1	Exponent
	GAD	1	1	G Exponent
	MXC	1	1	Microsequencer
	RAC	1	12	RAM Register Control
	SOP	4	2 3	Source Operand
L0213 (FBM, FMI FBox Multiply				
	MAA	4	9-10	Multiply ACC Extension
	MCR	1	11	Multiply Control B
	MCL	1	8	Multiply Carry Look Ahead
	MEP	4	1 2	Multiply Unpacker
	MPY	9	3-7	Multiply
	MXI	1	7	Multiply Extension
	MXU	1	12	Microsequencer
IDOX MCAs				
L0200 (IBB) IBox Buffer Decode				
	DBS	4	6	Double Buffer Slice
	IBF	9	1	Instruction Buffer
	IMC	1	A	IBox Flow Control
	IOP	4	-	IBox Op Bus
	SPD	1	9	Specifier Decode
	VAC	1	8	Valid Bit Logic
L0207 (ICA) IBox Control Logic A				
	JSI	1	C	Jsrall
	SEQ	1	5	Sequencer
	UCC	1	8	Micro-PC
L0208 (IDE) IBox Data Path				
	DEP	2	6	Data Path Parity
	IAD	6	1	IBox Address
	IVA	8	5	IBox Virtual Address
	OPA	1	7	Operand Pad Address
	UPK	1	6	Unpacker

Table 19-2 MCA Descriptions and Locations (Cont.)

Module Name	MCA Name	Qty	Location	Description
NBOX MCAs				
LG205 (MAP) NBox Address Path				
	ADA	5	1	Address Path A
	ADB	1	2	Address Path B
	CMA	2	4	Match
	MRG	x	5	Marksman
	MVP	1	6	Mailbox-Valid-Parity
LG204 (MCC) Control				
	ABS	1	4	ABus Control
	ARB	1	6	Arbitration
	CRA	1	A	Control Register A
	CRB	1	B	Control Register B
	ERR	1	E	Error
	MAT	1	8	Memory Array Interface
	MCP	1	C	Micro Control Function
	MMD	4	D	Memory Data Register
	MNS	2	1	MBox Microsequencer
	SES	1	9	Response
	STA	1	7	Status
LG204 (MLD) Data Path				
	CKP	1	F	Check Parity
	ECC	1	E	Error Correction Code
	MDP	1	1	MBox Data Path
	DTD	1	0	Data Registers
CLOCK MCA				
LG217 (CLK) Clock				
	CLC	1	3	Clock Control

†LG230 for VAX 8650
‡LG217 for VAX 8650

NOTE

"Location" refers to a specific module print, e.g., the Clock Control MCA is found on CLK1.

19.2 ECL TROUBLESHOOTING INFORMATION

The VAX 8600 processor logic is implemented with Emitter Coupled Logic (ECL) technology. There are numerous Technical Data books available which describe the theory of this family of logic, including the advantages and disadvantages of its use.

From a troubleshooting aspect, there are a number of concerns that must be considered when investigating ECL circuits.

19.2.1 Test Equipment

As ECL logic is relatively fast, comparable test equipment must be used. For scoping, a 200 MHz or faster oscilloscope is required, along with equal length probes and short (6 inch or less) ground clips. Beware that 'slinky' in the probe leads may distort the waveforms.

19.2.2 Termination

One of the most critical components in an ECL circuit is the termination. It can be the source of many circuit problems. The text that follows describes the basic ECL circuit and termination techniques used in the VAX 8600/8650, followed by a troubleshooting flowchart and related waveforms.

Every ECL signal run requires termination. The VAX 8600/8650 CPU employs parallel termination techniques on both single-ended and double-ended buses.

19.2.2.1 Single-ended ECL Signal Run - This type of termination is used to terminate a signal at or near the end of the run. The termination consists of a 50 ohm resistor tied to -2 V. See Figure 19-1.

19.2.2.2 Bi-directional ECL Signal Run - This type of termination (Figure 19-2) is used in bus applications where multiple bus drivers are employed, and the signal run is terminated at both ends of the run with 50 ohms to -2 V. For this application, 25 ohm bus drivers are used (as the circuit impedance will be 25 ohms). Note that these special 25 ohm bus drivers have a very high output impedance. When all drivers are disabled, the signal will be at a -2 V level (rather than a typical -1.8 V level).

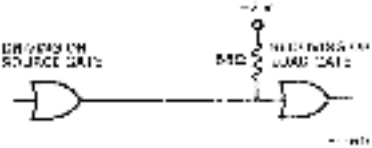


Figure 19-1 Single-ended ECL Signal Run

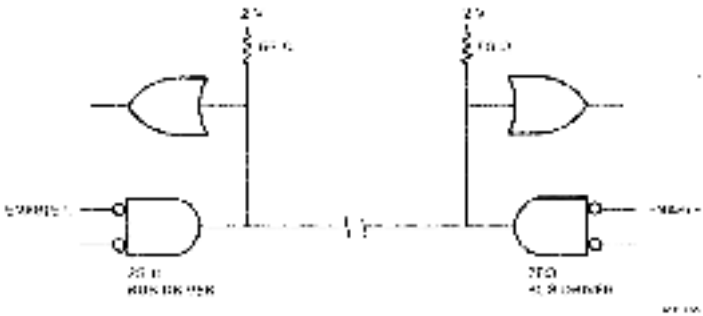


Figure 19-7 Bidirectional TTL Signal Bus

19.2.1.3 Termination Components - Following is a list of termination components used in the VAX 9000/6000 systems.

1. 56 ohm discrete component resistor (19-21602-06). For instance, A005 termination on STA/L0224 module (print SUM1)

2. 10-pin STR24 SIP (19-1749)-08) used on CPU modules where the signal does not leave the module.

3. 10-pin VTERM SIP (20-21247-01) used on CPU modules where the signal leaves the module or requires visibility.

Refer to the STR24 SIP and VTERM SIP data sheets for more details on these components.

19.2.2.4 How to Locate the Termination Point of a Given Signal - There are several methods used by the engineering CAD tool to indicate the termination of an ECL signal in the engineering drawings. Although there are no hard and fast rules to determine where a signal is terminated, there are some general guidelines that are used. Because that these guidelines are sometimes not followed.

1. If the signal does not leave the module on which it was generated, it will be terminated on that module. The schematic representation will depend upon whether the termination is on an STERM or a VTERM.

   If terminated by an STERM, a resistor will be drawn at the output of the signal source gate. If terminated by a VTERM, the signal is terminated within the VTERM which is shown on the visibility terminator print pages near the end of the module schematic set. See Figures 19-3 and 19-4. Figure 19-3 is taken from print set sheet XCOC, and Figure 19-4 is taken from print set sheet XCDV.

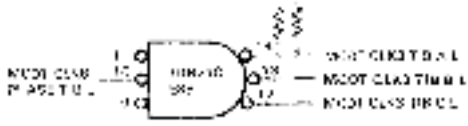


Figure 19-3 ST62M Termination.

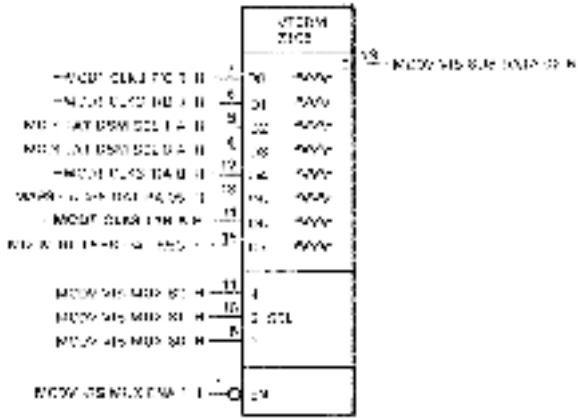


Figure 19-4 VT62M Termination

Notice two of the output signals shown on print XCLK, NL1Y CLK1 Y1.P & L and NCLK CLK1 Y1.N & L, are terminated on this module and do not have visibility (termination is shown at the gate outputs). The signal XCLK CLK1 Y1.P & L is terminated at the VT62M or gate array where NCLK is illustrated.

VT62M termination of a signal will identify the termination component location using the 'XAL' identifier within the SUB (xxxx). The physical type and location of the termination component for ST62M and discrete termination can be found only by using the module XAL2 SCHM wirelist.

2. If the signal leaves the module on which it was generated, the signal will be terminated on another module. To determine where the signal is terminated, use:

   a. the EL or MSMESCRI wirelist,

   b. the console SHOW XAL6 'signal name' command,

   c. the 'signal name sort' SDB table.

Once you determine which module the signal is terminated on, refer to the 'SDB visibility' pages (which are generally near the end of the module schematic set). This will give you the pin and component number of the termination package (either a VT62M 21P or a 10104V 7TP). See the example that follows.

Example 19-1 GIC Module Termination

The signal 'CSA UMCP 4 A E' (Figure 19-5) originates on print CSA9 and is terminated on print TCAK at VTERM X29 (Figure 19-6). Note that on print TCAK, the signal is listed '-CSA UMCP 4 A E' and there is no designator to show that the signal is on backplane pin A13-42. This is due to a deficiency in the CAD tools. Note that the wirelist NL sort entry gives the pin number of the backplane. As well as this, print page where the signal goes to (See Example 19-2). Since there is no designator for the signal on print TCAK, the print page number is excluded from the BL list. However, once you determine which module the signal feeds, you need only check the VTERM print page to find the signal.

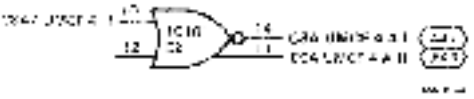


Figure 19-5 UMCP 4 A E Signal Generation

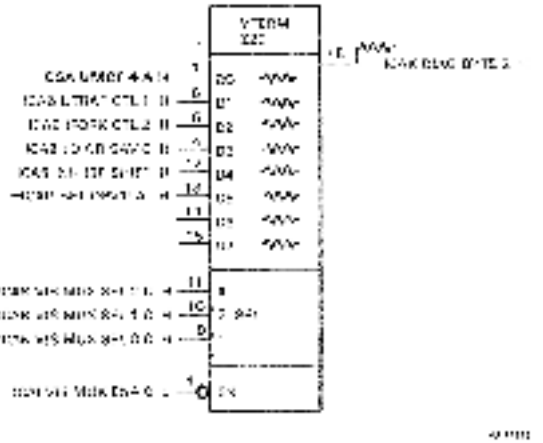


Figure 19-6 UMCP 4 A E Signal Termination

Example 19-2 Extract from CPU Backplane Wirelist BL-sort (70-19192)

```
-CSA UMCP 4 A E

AC2    A67  E0     0.00   0.00  -10.00    0.00    TCAK    IS    CSA8
AC3    A66                                        CSA4    C4
AC13   A42  E1P4   3.42  17.09   0.00  -200.00    TCAK    B7
AC15   A62                                        TCAK    D9
            E0I22  3.42  17.09 -10.00  -200.00
```

19.2.3 Troubleshooting ECL Signals

Figures 19-7 through 19-20 illustrate some of the more common ECL circuit failures that may be experienced. Although many of these problems were more spontaneous on older wire-wrapped backplanes, you may experience similar faults caused by electrical/mechanical failures due to current processes such as dirty finger pins, bad ground buses, nicked backplane wires (from an ECO), or cold solder joints. Figure 19-7 illustrates a typical ECL waveform.

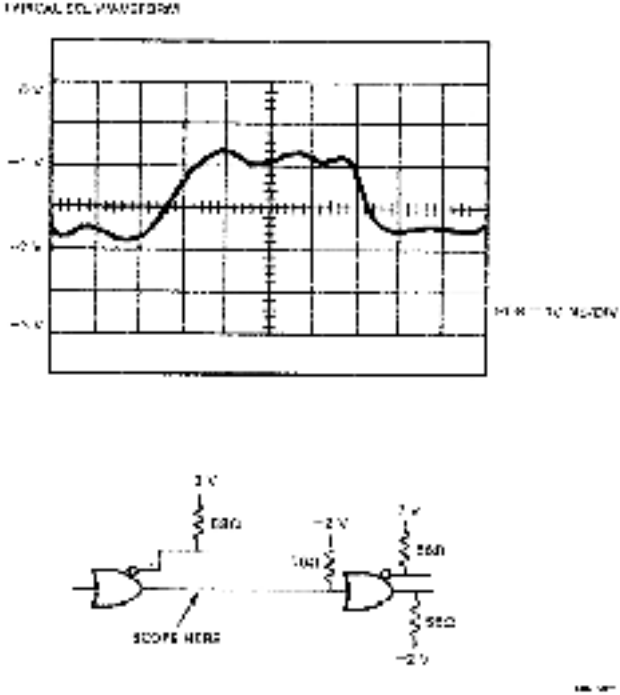


Figure 19-7 Typical ECL Waveform

Figure 19-8 Flowchart: SCI Signal Troubleshooting

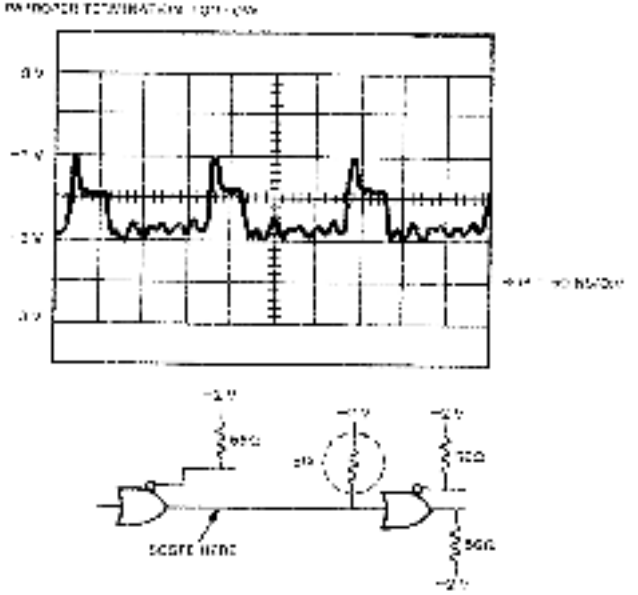


Figure 19-9 Improper Termination: Too Low

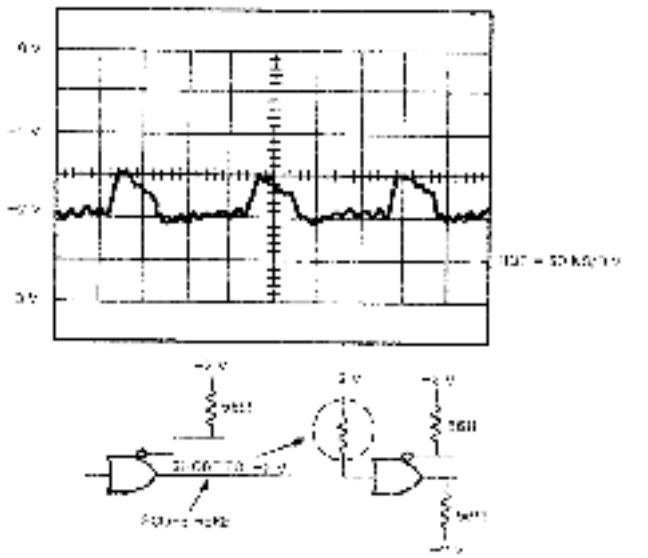


Figure 19-10 Output Shorted to -2 V

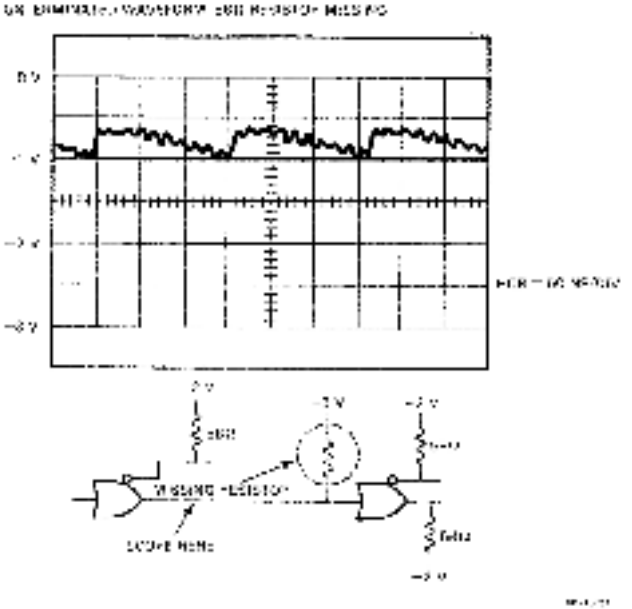


Figure 19-11 Concatenated Waveform, 56 Ohm Resistor Missing

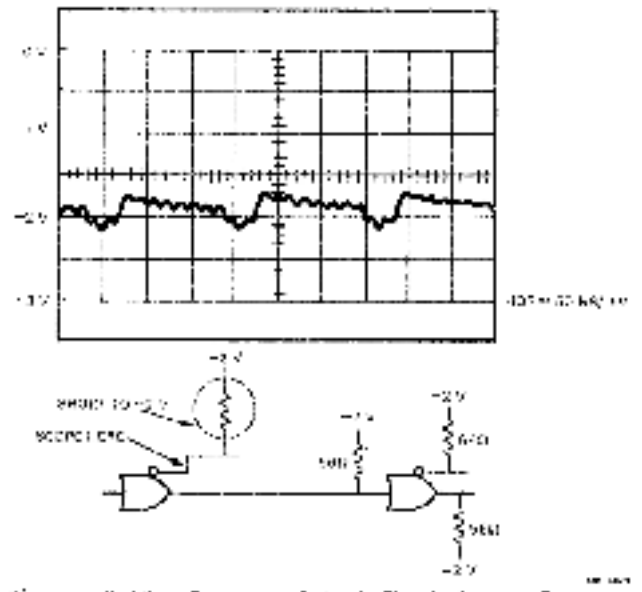


Figure 19-12 Inverter Output Shorted to - 2 v

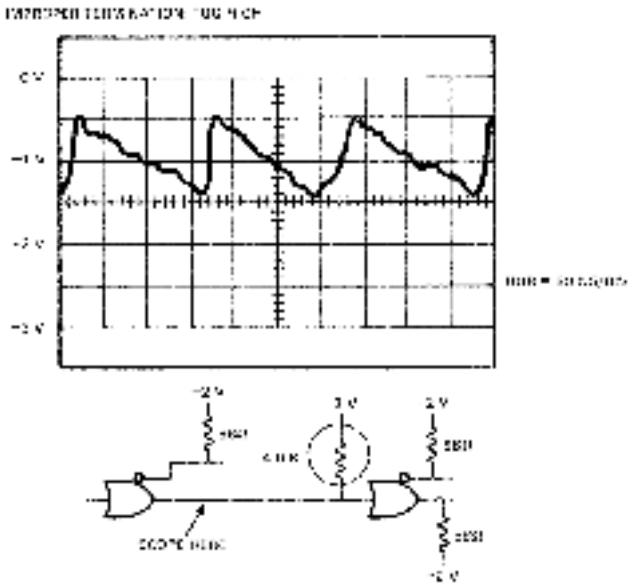


Figure 19-13 Improper Termination: Too High

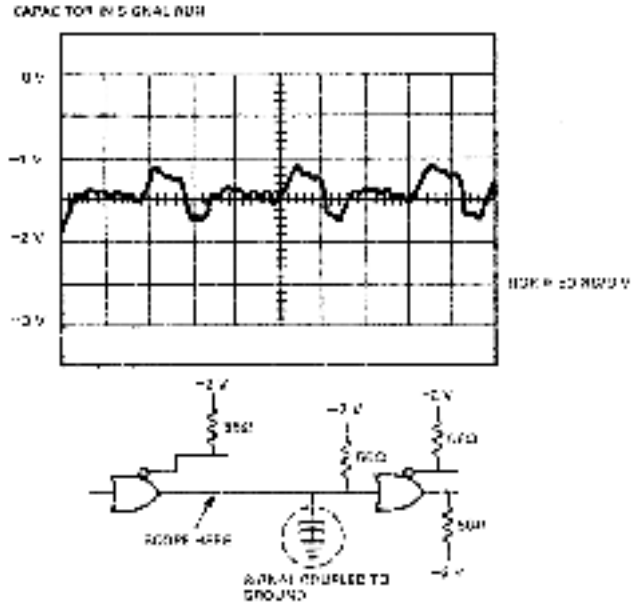


Figure 19-14 Capacitor in Signal Run

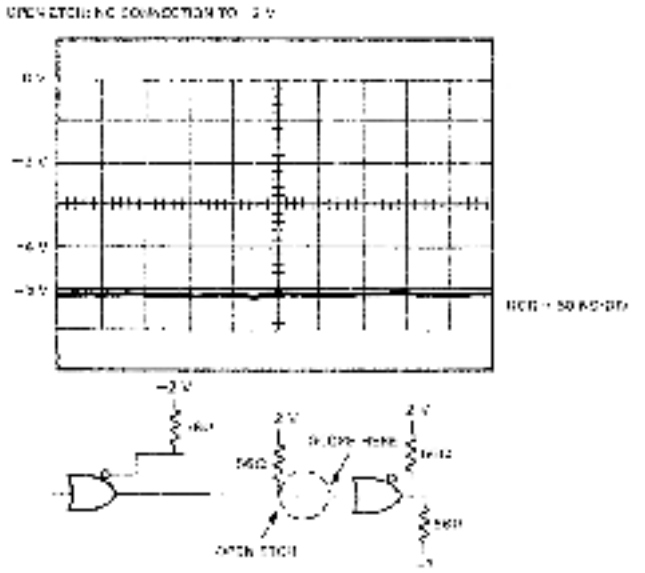


Figure 14-13. Open Risk, No Connection to - 2 V

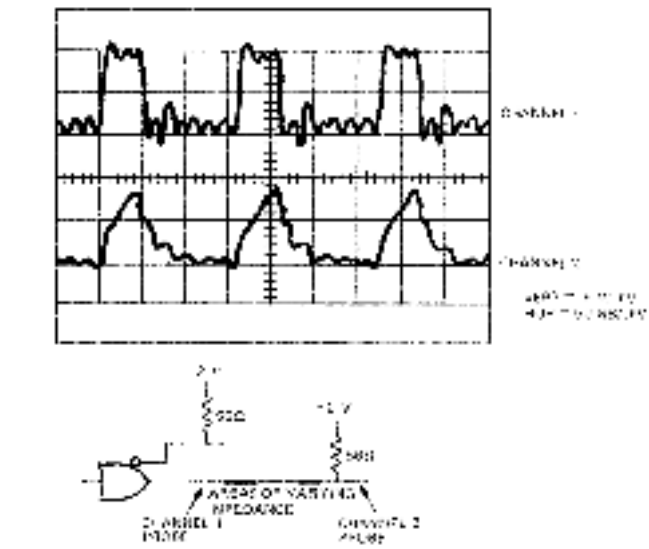


Figure 14-14. Impedance Mismatch

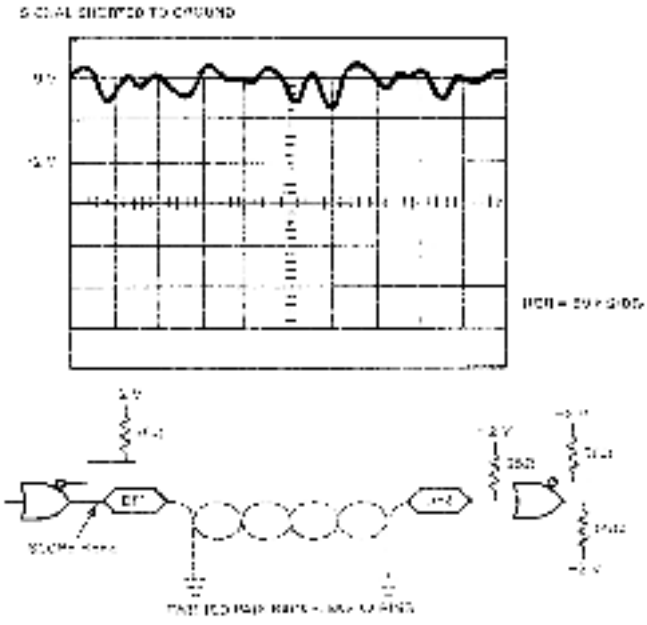


Figure 19-17 Signal Shorted to Ground

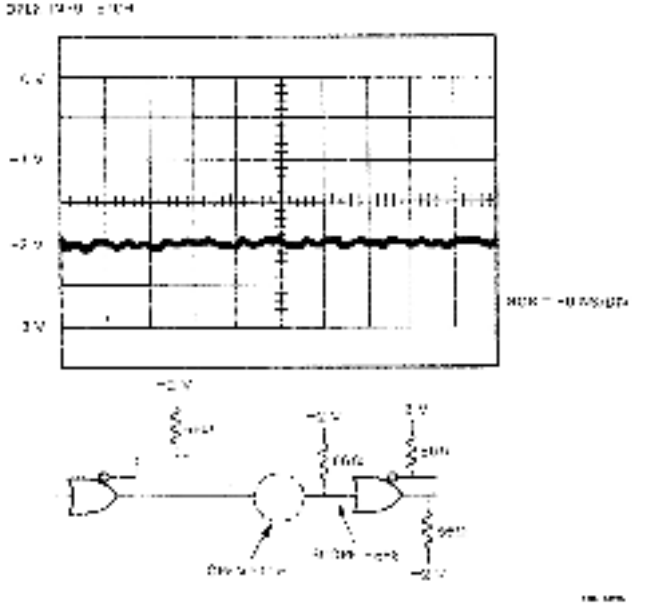


Figure 19-18 Open Input Bias

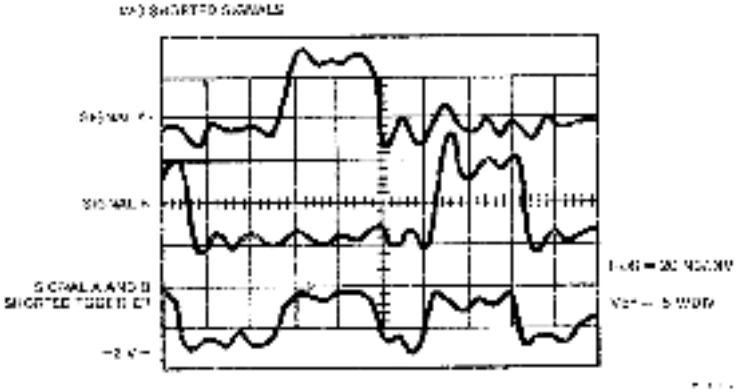


Figure 10-19 Two Shorted Signals

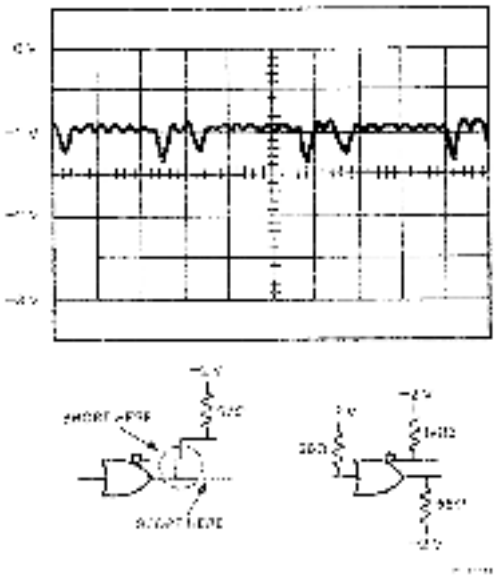


Figure 10-20 Complementary Outputs Shorted Together

CHAPTER 20

VAX 8600/8650 REGISTER DESCRIPTION

Chapter 20 has been reorganized. There are no longer multiple copies of any register. Unfavorable feedback from the field, the increase in page count due to size reduction, and additional material for some registers, necessitated the removal of duplicate registers.

The registers are listed alphabetically, except the CI780 registers, DW780 registers, DW730 registers, and SDTA registers are listed alphabetically within the individual groups, ie., All CI780 registers are listed under CI780.

In the index, the Internal Privileged Registers (IPRs), Internal Registers (IRs), Miscellaneous Registers (Misc), and Machine Check Stack Frame (MHCK Stack Frame) are listed both by individual registers and alphanetically within groups. The Stack Frame registers are listed in two sub-groups, Machine Check Stack Frame, and Stack Frame.

Keep in mind that the Stack Frame is not valid unless Error Handling Microcode has loaded the EBox scratch pad locations.

ACCS 20-7
ASTLVL 20-9

BTOXT0_3 20-8

CEUS REGISTERS
  BTOXU_3 20-8
  CMINT 20-70
  RBUFU_3 20-113
  RSUFT 20-114
  RXCS 20-119
  RXDB 20-130
  STXCS 20-176
  STXDB 20-177
  TURN 20-180
  TXCS 20-182
  TXDB 20-184
  UTCFG0_3 20-187
  XBUF0_3 20-189
  XBUFT 20-189

CI780 CNFGR 20-8
CI780 MADR 20-10
CI780 MDATR 20-10
CI780 PMCSR 20-13
CI780 PORT CONTROL 20-14
CI780 PORT ERR STATUS 20-16
CI780 PORT SAILING ADR 20-17
CI780 PORT PARAMETER 20-18
CI780 PORT QUE BLK BASE 20-18
CI780 PORT STATUS 20-19

CMTSC 20-20
CPC 20-20
CRDT 20-21
CSRS 20-22
CSWCTL 20-23
CBLINT 20-24
CSN MSACHS 20-26
CSWP 20-27

DCI 20-28
DIAGCS 20-144
DMAICA 20-164
DMATIP 20-165

DR750 CR READ 20-29
DR750 CR WRITE 20-30
DR780 DCBAR 20-31
DR780 DTE 20-32

DW780 BARVR4_7 20-33
DW780 BRSVS0_3 20-34
DW750 CNFGR 20-35
DW780 DCR 20-37
DW780 DPR0_12 20-38
DW780 DPRR 20-40
DW780 FAILED INTRSVS
      ADDRESS REGISTER 20-40
DW780 XBD_495 20-41
DW780 UAUR 20-42
DW780 USAR 20-45

DSCS 20-48
XBXNC1 20-51
XBANC2 20-51
XCMC 20-52
SBIFSR 20-53
XBMSTS 20-55
SBISC 20-60
SNI 20-63
SBRSTN 20-118
NSASAV 20-63
BSP 20-63
KSPA 20-64
BSPD 20-64
PVSQSAV 20-65

FSxSUR 20-66

InSCR 20-68
IBSDR 20-70
ICCS 20-71
ICR 20-72
IPL 20-72

INTERNAL PRIVILEGED REGISTERS
  ACCS 20-7
  ASTLVL 20-7
  CRBT 20-21
  CSWP 20-27
  DFI 20-28
  KSUR 20-60
  BSP 20-63
  KSPA 20-64
  USPD 20-64
  ICCS 20-71
  ICR 20-72
  IPL 20-72
  TSP 20-73
  KSP 20-73
  MAPEN 20-80
  MCCTL 20-80
  MDCTL 20-86
  MDECC 20-87
  MEMA 20-90
  MEMB 20-91
  MICR 20-98
  PCB2 20-98
  PCER 20-98
  PLB2 20-99
  PLCR 20-99
  PAMACC 20-100
  PAMLOC 20-101
  PCRR 20-102
  PRR 20-104
  PTK 20-107
  RXCS 20-129
  RXDB 20-130
  SID 20-166
  SCAH 20-166
  SID 20-171
  SIAR 20-173
  SIDR 20-173

MINIMAL PRIVELEGED REGISTERS
(continued)
ESP 20-174
SSP 20-175
USP 20-176
SCXDB 20-177
IDLTM 20-177
ISLA 20-178
IPIS 20-178
IODR 20-179
PXIS 20-182
IXDB 20-184

INTERNAL REGISTERS
CPC 20-20
CSDS 20-22
PSWCTL 20-23
CSLINT 20-24
EDCS 20-48
EDMC 20-52
EDPSR 20-53
SNG 20-63
SSASAV 20-63
IRESR 20-68
ISASAV 20-72
IVASAV 20-73
MEAR 20-89
MEDR 20-89
MSYAT1 20-93
MSTAT2 20-96
VISASAV 20-108
VECBITS 20-108

IBASAV 20-72
ISP 20-73
IVASAV 20-73

KSP 20-73

LRDR 20-74
LRSR 20-74
LTHR 20-75
LMCR 20-76
LMHR_HI_ORDER 20-78
LMHR_LO_ORDER 20-77

MACHINE CHECK STACK FRAME
CPC 20-20
CSDS 20-22
CSHCTL 20-23
CSLINT 20-24
EDCS 20-48
EDXMD1 20-51
EDXMD2 20-51
EDPSR 20-53
KSHSTK 20-65
SSASAV 20-63
SVMOSAV 20-65
PSXSHK 20-66
IRESR 20-68
ISASAV 20-72

IVASAV 20-73
KSPCC 20-82
MEAR 20-89
MEDR 20-89
KMMR1 20-91
MSTAT1 20-93
MSTAT2 20-96
SP 20-101
PSL 20-105
SPHOLD 20-170
VIBASAV 20-180

MATMT 20-154
MAPEN 20-80
MCCTL 20-80
MCERG 20-81
MCSR1 20-82
MCSR2 20-83
MCSR3 READ 20-84
MCSR3 WRITE 20-85
MDCTL 20-86
MDKLU 20-87
MEAR 20-89
MEBD 20-89
MENA 20-90
MERG 20-91

MISCELLANEOUS REGISTERS
SVMOSAV 20-65
IBGPR 20-70
PSL 20-105
SPADS 20-174
STATE 20-175

MSYAT1 20-93
MSTAT2 20-96

NICR 20-98

PCI REGISTERS
ESCR 20-76
KRHR 20-74
ESMR 20-77
ESBR 20-74
ETHR 20-75
ERCR 20-76
KMHR 20-77
LRCR 20-74
LRHR 20-74
LRLR 20-77
LRSR 20-74
LTHR 20-75
LMCR 20-76
LMHR_HI_ORDER 20-78
LMHR_LO_ORDER 20-77
RMSR 20-113
RXCR 20-76
RRHR 20-74
RMHR 20-77
RRBR 20-74

PCI REGISTERS
(continued)
RTHR 20-76
RMCR 20-76
RMHR 20-77

PCBR 20-98
PCLR 20-96
PIBR 20-99
PILR 20-99
PAMACC 20-100
PAMLOC 20-101
PC 20-101
PCBB 20-102
PECAD 20-105
PERAB 20-103
PER 20-104
PSL 20-105
PTB 20-107

QADR0 20-108
QADR1 20-108
QCER0 20-109
QCER1 20-110
QCER2 20-111
QCER3 20-112
QDATA0 20-113
QDATA1 20-113
QUADCLR 20-159

RHSIC 20-113
REGPU_3 20-113
RHHVC 20-114

RH780 HCR 20-114
RH780 CAR 20-115
RH780 DR 20-115
RH780 CSR 20-116
RH780 DR 20-117
RH780 SVR 20-118
RH780 SR 20-118
RH780 VAR 20-121

RLSA 20-121
RLCSR 20-122
RLDA GET STATUS 20-121
RLDA READ WRITE 20-125
RLDA SEEK 20-126
RLMPR GET STATUS 20-126
RLMPR READ 20-128
RLMPR WRITE 20-128
RXCS 20-129
RXDB 20-130

SBIA CR 20-142
SBIA CSR 20-143

SBIA REGISTERS
SBIA CONFIG REG 20-142
SBIA CONTROL STATUS 20-141
SBIA DIAG CONTROL 20-144
SBIA ERR CA 20-146
SBIA ERR ID 20-147
SBIA ERROR SUMMARY 20-140
SBIA SBI ERROR 20-152
SBIA SBI FAULT STATUS 20-154
SBIA SBI MAINT 20-156
SBIA SBI QUADCLR 20-150
SBIA SBI SILO 20-159
SBIA SBI TIMEOUT ADR 20-160
SBIA SBI TIMEOUT ADR 20-164
SBIA SBI UNJAM 20-161
SBIA SBI VECTOR 20-165

SBIERR 20-152
SBISTS 20-154
SBR 20-166
SCBB 20-106
SCON 20-167
SDDG 20-168
SDMG 20-169
SFBCNT 20-170
SID 20-171
SILO 20-159
SILOCOMP 20-161
STRF 20-173
SIBR 20-175
SLR 20-174
SVADR 20-174
SBP 20-175

STACK FRAME
CPC 20-20
CSES 20-22
CSHCTL 20-23
CSLINT 20-24
CSCS 20-48
EBXWDJ 20-51
EBXWJ2 20-51
EDPSR 20-53
EHMST2 20-55
ESASAV 20-62
EVMQSAV 20-65
FPXFER 20-66
IDBBR 20-68
ISASAV 20-72
IVASAV 20-73
KDECC 20-87
KMAR 20-89
KEDR 20-89
KKMG 20-91
KSTAT1 20-93
KSTAT2 20-96
PC 20-101
PSL 20-105
SFBCNT 20-170
VIBASAV 20-186

STATR 20-175
STXCS 20-176
STXDB 20-177

TMCHK 20-177
TBIA 20-170
IBIS 20-178
TOSOR 20-164
TODR 20-179

TOY REGISTERS
TRDR 20-179
TRER 20-180
TWCR 20-181
TWDR 20-179

TRDR 20-179
TBSR 20-180
TCAR 20-180
TNCR 20-181
TMDR 20-179
TXCS 20-182
TXDB 20-184

UNJAM 20-164

USART REGISTERS
BRCR 20-76
BRSR 20-74
BRMR 20-77
CMSR 20-78
RTHR 20-75
RMIR 20-76
BMMR 20-77
LSCR 20-76
LRHR 20-74
LRMR 20-77
LTHR 20-74
LTIR 20-75
SWCR 20-76
LWMR_HI_ORDER 20-78
LWMR_LO_ORDER 20-77
PBSR 20-213
XMCR 20-76
RGSR 20-74
RSMR 20-77
RRSR 20 74
RTHR 20-75
RMCR 20-76
RMMR 20-77

USP 20-197
UTOY0_3 20-187

VECTOR 20-165
VIBASAV 20-185
VPCDIIS 20-188

XBGP0_3 20-189
XBUFC 20-189



Notes: The ACCS is a read/write register that contains the accelerator type, the revision status, and the enable bit.

<31:24> RESERVED

<23:16> FBOX REVISION
Number for the VAX 8600 FBox, or for the FPA and DJA boards that replace the FBox when there is no FBox present.

<15> FBOX ENABLE
Controls the execution of Floating Point instructions by the FBox. Writing a one to this position initializes the FBox and enables it to execute VAX instructions.

<14:08> RESERVED

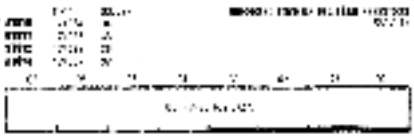
<07:00> FBOX TYPE
Specifies the type of the accelerator. For a VAX 8600 FBox, this field is set to a 1 by KF86K (operation location: 7C). If no accelerator is present, this field equals 0.



<31:03> MBZ
Must be zero.

<02:00> ASYNCHRONOUS SYSTEM TRAP LEVEL
Contains access mode number (established by software) of the most privileged access mode for which an Asynchronous System Trap (AST) is pending. Controls the triggering of the AST delivery interrupt during REI instructions.

ASTLVL	MEANING
0	AST pending for access mode 0 (KERNEL)
1	AST pending for access mode 1 (EXECUTIVE)
2	AST pending for access mode 2 (SUPERVISOR)
3	AST pending for access mode 3 (USER)
4	No pending AST
5-7	Reserved for Digital

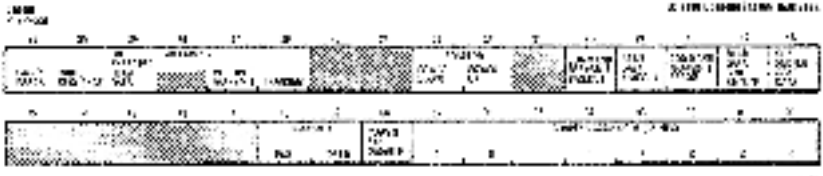


The BTOY registers consist of four EBus RAM byte locations.

<07:00> BUFFERED TOY DATA
The console software loads the BTOY registers from the TOY counter once every 10 milliseconds. This maintained value is read by EBox microcode when executing an MFPR #^X08 instruction. The byte alignment is:

<31-24> = BTOY 3
<23-16> = BTOY 2
<15-08> = BTOY 1
<07-00> = BTOY 0

NOTE: Refer to the VAX 8600/8650 Console Technical Description Manual, EK-KA86C-TD, for more detailed information on TOY CLOCK operation.



<31:26> SBI FAULT BITS
These bits are set when the port detects the respective fault condition as described below. The fault bits are read only.

<31> PARITY FAULT
Set when the port detects an SBI Parity error.

<30> WRITE SEQUENCE FAULT
Set when the port receives a write mask command that is not immediately followed by the expected write data.

<29> UNEXPECTED READ DATA FAULT
Set when the port receives read data but did not issue a read command.

<28> RESERVED

<31> MULTIPLE TRANSMIT FAULT
Set when the 10 bits transmitted by the port do not match the 10 bits received back from the SBI.

<26> TRANSMIT FAULT
Set if the CI780 was the SBI master that caused the SBI FAULT line to assert.

<25:24> UNPREDICTABLE

<23> POWER DOWN
Set if the port is powering down. PDN is set by the assertion of SUPPLY ADAPTER if the port is in the uninitialized state, otherwise it is set by the microcode via the PDN bit. Cleared by writing a 1 to it or by setting the PUP bit.

<22> POWER UP
Set by the negation of SUPPLY ADAPTER. Cleared by writing a 1 to it or by setting the PDN bit.

<21> RESERVED

<20> COMMAND TRANSMIT TIMEOUT
Set when the port initiates an SBI transfer and does not receive an ACK or error confirmation within 102 microseconds. The timeout period is selectable by backplane jumpers.

<19> READ DATA TIMEOUT
Set when the port initiates an SBI read transfer and read data is not returned within 102 microseconds.

<18> COMMAND TRANSMIT ERROR
Set when the port receives an error confirmation in response to a port initiated SBI command transmission.

<17> READ DATA SUBSTITUTE
Set when the port receives an RDS (unpredictable read data) confirmation in response to a port initiated SBI read command.

<16> CORRECTED READ DATA
Set when the port receives a CRD confirmation in response to a port initiated SBI read command.

<15:11> RESERVED

<10> TRANSMIT FAIL
Set by the microcode through the miscellaneous control field when PFV (power fail valid) and ASSERT FAIL are true.

<09> TRANSMIT DEAD
Set by the microcode through the miscellaneous control field when PFV and ASSERT DEAD are true.

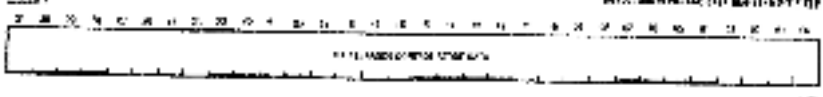
<08> POWER FAIL DISABLE
Set when the SBI FAIL and SBI DEAD drivers to the SBI are disabled.

<07:00> ADAPTOR CODE
These bits contain the CI780 SBI adapter code, 38(16).



<31:1]> RESERVED

<12:00> MADR
Contains the address of the control store location to be accessed. It is read or written only in the uninitialized state.



The CI780 MDATR (maintenance data register) does not exist as a physical register. A read or write of MDATR will read or write the microword in the control store location specified by the address in the MADR (maintenance address register). When MADR 12 = 0, MDATR (31:00) contains microword bits (31:00). When MADR 12 = 1, MDATR (15:00) contains microword bits (47:32) (MDATR (31:16) are all 0's). MDATR is read or written only in the uninitialized state.

<47> SYNC
A programmable bit that is used during user debugging to indicate the execution of a specific microword. The SYNC bit is not included in the parity check of the microword. The SYNC bit can be written in both the RAM and PROM areas of the CS (control store). The bit is available on the port backplane.

<46> PARITY
The odd parity bit on bits (63:00) of the CS microword.

<45:43> ALU FUNCTION <2:0>
Function code for the 2901 ALU on the DP (data path).

<42:40> ALU SOURCE CODE <2:0>
Operand source code for the 2901 ALU on the DP.

<39:37> ALU DESTINATION CODE <2:0>
Destination code for the 2901 ALU on the DP.

<36:33> ALU A/B ADDRESS LINES <3:0>
The A and B address lines for the 2901 scratch pads on the DP.

<32> TYPE
Selects the definition of bits <31:24> as shown below.

<31:24> LITERAL <7:0>
Valid when TYPE = 0. Used in the DP as a number or as an address.

<31:24> LINK AND PB (PACKET BUFFER) CONTROL BITS
Valid when bit 32 = 1. The bit fields are defined below.

<31> RESERVED

<30> SELECT
Indicates that the LINK CONTROL bits (27:24) are valid.

<29:28> PBUX <1:0>
Selects a byte in the packet buffer input and output registers on the DP.

<25:24> LINK CONTROL <3:0>
Specifies operations on the link and PB. This field is valid when SELECT = 1.

^<23:21> IN SOURCE <2:0>
Selects the source of BUS IN (internal bus) data in the DP.

^<20:17> IN DESTINATION <3:0>
Selects the destination for BUS IN data in the DP.

^These bits bypass the microword register and go directly to the DP.

<16:12> SEQUENCE CONTROL <4:0>
Specifies the operation of the 2911 micro-sequencer, selects the branch conditions that alter the microaddress, and selects the definition of bit <11:00>.

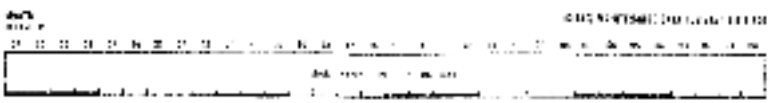
<11:00> NEXT MICROADDRESS
This field is the base address that is modified by the branch bits to form the address of the next microword. It allows the microcode to jump to any address in the CS. This field is valid as long as the Sequence Control field is not all 1's.

<11:00> MISCELLANEOUS CONTROL
This field (MISCELLANEOUS CONTROL) allows the microcode to control miscellaneous flags and functions in the port. The field is valid when the Sequence Control field is all 1's. The Miscellaneous Control bits are described below.

<11> MCLR
This bit (MAINTENANCE CLEAR) causes the port to enter the uninitialized state.

<10> INTERRUPT REQUEST
Sets the INTERRUPT REQUEST flag that initiates an interrupt sequence to the host CPU.

<09> INITIALIZE
Generates an INITIALIZE signal to the link.



<05> CLEAR Registers WRITE
Clears the Register Write Diag in the CP.

<07> POWER Fail VALID (PFV)
When the POWER-FAIL VALID bit is set, the ASRT [BAD and ASXI FAIL bits are valid.

<06> ASRT BAD
Periitated processor initialization and booting.

<05> ASXI FAIL
Facilitates programmer initialization and boot ng.

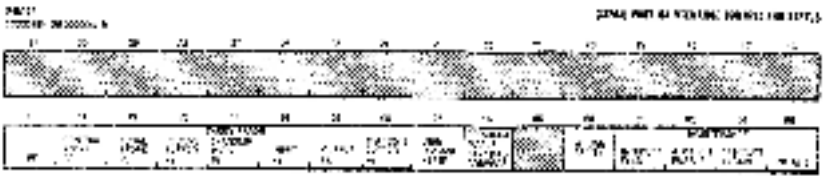
<04> SET A GO
Starts an external bus transfer with the host using the A parameters.

<03> SET B GO
Starts an external bus transfer with host using the B parameters.

<02> UP POWER DOWN
Allows the microcode to set the Power Down bit in the port Configuration register.

<01> INHIBIT PARITY
Set during a UP read of the first byte from a packet buffer. The first byte read is always undefined data. INH PARITY receive buffer parity error) prevents a parity error from occurring on the undefined data.

<00> RESERVED



<15> PARITY ERROR
This bit is set when any of bits <14:08> are set. It is cleared when <14:08> bits are all cleared.

<14> CONTROL STORE PARITY ERROR
This bit is set when a parity error is detected in the Control Store in the Packet Buffer (PB). Note: CSPE can only be set when the microcode is running.

<13> LOCAL STORE PARITY ERROR
This bit is set when a parity error is detected while reading the Local Store (LS) or the Virtual Circuit Descriptor table (VCDT). Note: A Local Store Parity Error can only be set by a microcode read of the LS or the VCDT. It will not get during an unsolicited SBI request.

<12> RECEIVE BUFFER PARITY ERROR
This bit is set when a parity error is detected while reading a packet buffer.

<11> TRANSMIT DATA PARITY ERROR
This bit is set when a parity error is detected in the link transmit channel.

<10> INVLT PARITY ERROR
This bit is set when a parity error is detected on a data transfer from the SBI module to the PB.

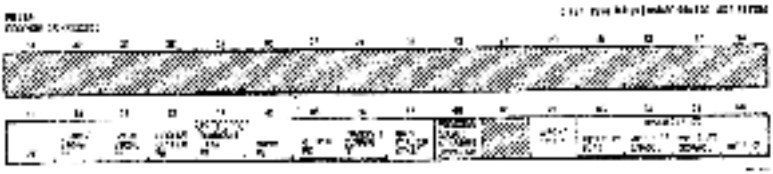
<09> OUTPUT PARITY ERROR
This bit is set when a parity error is detected on a data transfer from the output buffer to the transceivers within the SBI module.

<08> TRANSMIT BUFFER PARITY ERROR
This bit is set when a parity error is detected while the PB is unloading a transmit buffer.

<07> UNINITIALIZED STATE
This bit is set when the port is in the uninitialized state. Note: The microcode is not running and the port will not respond to data packet traffic. UNINIT is set by DEAD, MIM (maintenance initialize) or MTE (maintenance error). The microcode is started when UNINIT is cleared by writing a 1 into the Port Initialize Control Register (PICR) or by a boot time-out.

<06> PROGRAMMABLE STARTING ADDRESS
When set, the microcode will start at the address in the MADR (maintenance address register), when a "1" is written into the PICR or a boot time-out occurs. When reset the microcode starts at location 000.

<05> RESERVED



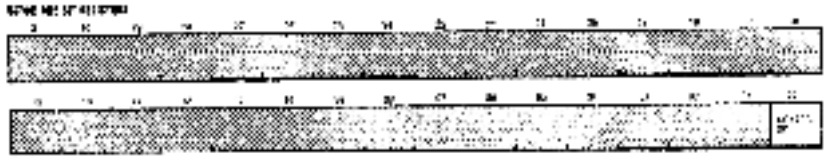
<Q4> WRONG PARITY
This bit is set when the SP (data path) parity generator/checker will generate and check even parity instead of odd. Note: This bit is used to generate parity errors for maintenance purposes.

<Q3> MAINTENANCE INTERRUPT PLAC
This bit is set when an interrupt-causing condition has occurred.

<Q2> MAINTENANCE INTERRUPT ENABLE
This bit is set by PC DC 00 from the SBI module or by writing this with a "1". Note: When set, interrupts are enabled.

<Q1> MAINTENANCE TIMER DISABLE
This bit is set when the port and maintenance timers are disabled and cannot cause an interrupt. When reset, the timers are enabled.

<Q0> MAINTENANCE INITIALIZE
This bit is set when an initialize signal is generated that clears all port errors and leaves the port in the uninitialized state.



The port control registers are 32-bit write only registers, and are written by the port driver. Each port control register controls one function, and this function is invoked when a "1" is written to the register. The register offset, name and function, and description follows.

XXXXX900 PORT COMMAND QUEUE 0 CONTROL REGISTER (PCQ0CR)
When the port driver inserts an entry in an empty CMDQ0, the port driver writes PCQ0CR to initiate port execution of the command queue. PCQ0CR can be written only when the port is in the enabled or enabled/maintenance state.

XXXXX90C PORT COMMAND QUEUE 1 CONTROL REGISTER (PCQ1CR)
Same as PCQ0CR except refers to CMDQ1.

XXXXX910 PORT COMMAND QUEUE 2 CONTROL REGISTER (PCQ2CR)
Same as PCQ0CR except refers to CMDQ2.

XXXXX914 PORT COMMAND QUEUE 3 CONTROL REGISTER (PCQ3CR)
Same as PCQ0CR except refers to CMDQ3.

XXXXX918 PORT STATUS RESPONSE CONTROL REGISTER (PSRCR)
After the port driver has received an interrupt and read the PSR, it returns the PSR to the port by writing PSRCR.

XXXXX91C PORT ENABLE CONTROL REGISTER (PECR)
The port driver enables the port by writing PECR. PECR is ignored if the port is in the uninitialized, uninitialized/maintenance, disabled, or enabled/maintenance states.

XXXXX920 PORT DISABLE CONTROL REGISTER (PDCR)
The port driver disables the port by writing PDCR. When the port is disabled, the port requests an interrupt. PDCR is ignored if the port is in the uninitialized, uninitialized/maintenance, disabled, or disabled/maintenance state.

XXXXX924 PORT INITIALIZE CONTROL REGISTER (PICR)
The port driver initializes the port by writing PICR. When the initialization is complete the port requests an interrupt. As part of the initialization, the maintenance timer is set to expire in 100 seconds.

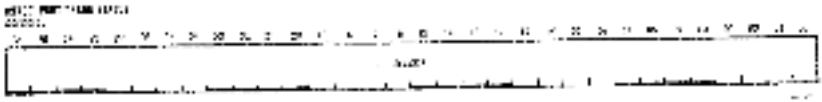
XXXXX928 PORT DATAGRAM FREE QUEUE CONTROL REGISTER (PDFQCR)
When the port driver inserts an entry on the DFREEQ and the latter was previously empty, the port driver writes PDFQCR to indicate the availability of DFREEQ entries. PDFQCR can be written only if the port is in the enabled or enabled/maintenance state.

XXXXX92C PORT MESSAGE FREE QUEUE CONTROL REGISTER (PMFQCR)
Same as PDFQCR except refers to MFREEQ.

XXXXX930 PORT MAINTENANCE TIMER CONTROL REGISTER (PMTCR)
The port driver forces the maintenance timer to reset its expiration time by writing the PMTCR. If the PMTCR is not written again before the expiration time, the port will enter the uninitialized/maintenance state and request a maintenance timer expiration interrupt (port status chg). PMTCR is ignored if the maintenance timer is not running.

XXXXX934 PORT MAINTENANCE TIMER EXPIRATION CONTROL REGISTER (PMTECR)
The port driver forces a maintenance timer expiration interrupt by writing the PMTECR. This register may be written only when the port is in the enabled, enabled/maintenance, disabled, and disabled/maintenance states and only while the maintenance timer is not disabled.

VAX 8600/8650 REGISTER DESCRIPTION
CI780 PORT SLAVE STATUS



The Port Error Status Register (PESR) indicates the type of error which resulted in a DSE interrupt. The PESR is read only by the port driver and valid after a DSE interrupt. The codes and a brief description follow:

CODE DESCRIPTION

1. Illegal system virtual address format. Bits <31:30> of the virtual address was not equal to. The Port Failing Address Register (PFAR) contains a virtual address.

2. Non-existent system virtual address. VA <29:9> >= SPT_LEN. PFAR contains a virtual address.

3. Invalid system PTE. Bits <31,26,22> not equal 1xx, 000, or 001. PFAR contains the virtual address being mapped.

4. Invalid buffer PTE. Bits <31,26,22> not equal 1xx, 000, or 001. PFAR contains the PTE virtual address.

5. Non-existent system global virtual address. GPIX >= GPT_LEN. PFAR contains the virtual address.

6. Non-existent buffer global virtual address. GPIX >= GPT_LEN. PFAR contains the VAB virtual address.

7. Invalid system global PTE. PTE <31,26,22> not equal 1xx or 000. PFAR contains the virtual address.

8. Invalid buffer global PTE. PTE <31,26,22> not equal 1xx or 000. PFAR contains the PTE virtual address.

9. Invalid system global PTE mapping. The system virtual address of system global PTE is invalid globally mapped. PFAR contains the virtual address.

10. Invalid buffer global PTE mapping. The system virtual address of buffer global PTE is invalid globally mapped. PFAR contains PTE virtual address.

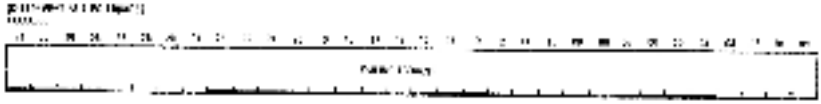
11. Queue interlock retry failure. Interlock was busied and found locked an implementation specific consecutive number of times. PFAR contains the queue head virtual address.

CODE DESCRIPTION

12. Illegal queue interlock alignment. FLINK <2:1> or BLINK <2:1> is not 0. PFAR contains the queue head virtual address.

13. Illegal SCS format. A field of the SCB specified to be zero was found to be other than zero. (The check for zero fields is optional so not all ports will return this error.) PFAR contains the field offset in the SCB in bytes.

14. Register protocol violation. Register was written with the wrong value or under the wrong conditions. (The check for protocol violations is optional so not all ports will return this error.) PFAR contains the register byte offset from device base address.

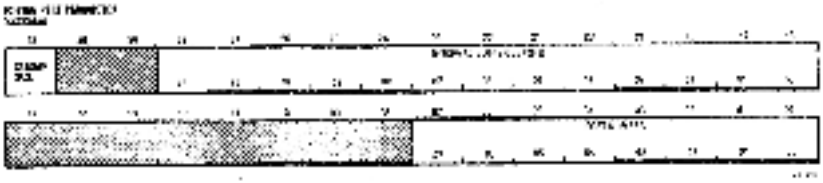


After a PDR or DSE interrupt and after a response with buffer memory system error status, the Port Failing Address Register (PFAR) contains the memory address at which the failure occurred. The address may be the exact failing address, an address in the same page as the failing address, or, in the case of DSE interrupts, an address in some part of the data structure. For DSE interrupts PFAR contains a virtual address or offset, while for MSR interrupts and buffer memory system errors the PFAR contains a physical address.

Since the port continues command execution and packet processing after buffer memory system errors occur, the PFAR is overwritten if subsequent errors occur. For PDR and MSR interrupts the PFAR is effectively fixed since the port enters the disabled or disabled/maintenance state.

PFAR is read only by the port driver and readable after a DSE or MSR interrupt or after a response with buffer memory system error status.

VAX 8600/8650 REGISTER DESCRIPTION
CI780 PORT PARAMETER REGISTER (PPR)
CI780 PORT QUEUE BLOCK BASE (PQBBR)

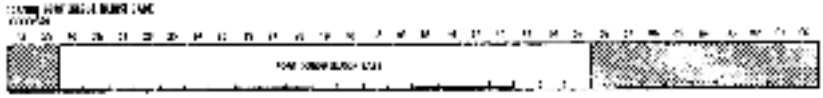


The Port Parameter Register (PPR) contains port implementation parameters and the port number. The value of the PPR is set by the port during initialization and valid after a DIC interrupt. PPR is read only by the port driver.

<31> CLUSTER SIZE
Cluster size = 0 indicates a maximum of 16 ports on the CI. Cluster size = 1 indicates a maximum of 256 ports on the CI.

<28:16> INTERNAL BUFFER LENGTH
The internal buffer length indicates the size of internal buffers available for message and data transfers. The maximum data packet is IBUF_LEN - 16 bytes. Maximum message or datagram length = IBUF_LEN. The minimum value is 592.

<07:00> PORT NUMBER

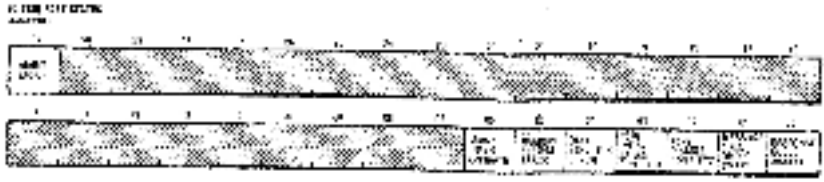


This register contains the physical address of the base of the Port Queue Block. The PQBB register is read/write by the port driver and writable only when the port is in the disabled or disabled/maintenance state.

<31:30> MBZ

<29:09> Port Queue Block Base Address

<08:00> MBZ



This register returns status to the port driver after an interrupt. When an interrupt is requested by the port, the value of the Port Status Register (P9S) is fixed and is not changed until the port driver releases the register by writing the Port Status Release Control Register (P9RCR).

<31> MAINTENANCE ERROR
This bit, if set, indicates that the port has detected an implementation specific error (or hardware status condition). The cause of the error may be more accurately determined from the implementation specific maintenance registers.

<30:07> RESERVED

<06> MAINTENANCE TIMER EXPIRATION
When this bit is set, it indicates that the maintenance timer has expired. The port is in the uninitialized/maintenance state.

<05> MEMORY SYSTEM ERROR
When Memory System Error is set, the port has encountered an uncorrectable data or non-existent memory error while referencing memory. The port is in the disabled or disabled/maintenance state. See the Port Failing Address Register for further information.

<04> DATA STRUCTURE ERROR
When this bit is set, the port has encountered an error in a port data structure (i.e., queue entry, PCB, BDT or page table). The port is in the disabled or disabled/maintenance state. See Port Error Status Register and the Port Failing Address Register for further information. The errors in the queue structure leave the queues locked.

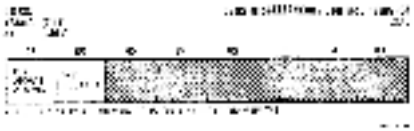
<03> PORT INITIALIZATION COMPLETE
When Port Initialization Complete is set, the port has completed internal initialization. The port is in the disabled or disabled/maintenance state.

<02> PORT DISABLE COMPLETE
When Port Disable Complete is set, the port is in the disabled or disabled/maintenance state.

<01> MESSAGE FREE QUEUE EMPTY
When set, Message Free Queue Empty indicates that the port attempted to remove an entry from the message free queue and found it empty. Port processing of commands continues as the message free queue may not be empty at the time the port driver gets control.

<00> RESPONSE QUEUE AVAILABLE
When set, Response Queue Available indicates that the port has inserted an entry in an empty response queue.

VAX 8600/8650 REGISTER DESCRIPTION
CMISC
CEC



This register is implemented by two flip-flops outside of the Mbox RAM. It contains two bits: TOY Update Pending (TOYPU) and T-11 Restart (T11RS).

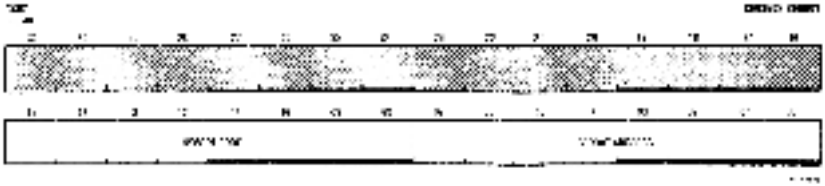
<0:0> TOY UPDATE PENDING
CLIS mnem n
This bit is set by the console software while it is updating the BTOY register from the TOY register. This operation is performed every 10 ms by the console software.

<1:1> T-11 RESTART
CLIS mnem n
Setting this bit forces a console (P-11) interrupt to occur, as which reboots the console from PROM. This bit may or may not be set by the MACRO program running in the CPU. In other words, this is done by writing the Console Reboot (CRBT) IPR. The console program may also select this bit individually by writing MCSR3 <27:16> with a function code of 7 which will in so force a T-11 reboot via PROM.

<5:3> MBZ/RAZ/RAO



<31:30> CURRENT PROGRAM COUNTER
This register contains the address (Macro PC) of the instruction that the IBox Address Calculation Unit will process next.



This write only register is used by software to reboot the console processor without having to reboot the whole system. To determine when to reboot the console, microcode examines the TOY every 90 sec. to see if the system has been updated by the console software. If TOY hasn't changed, the console needs to be rebooted. The reboot Code/Address is supplied by microcode not VMS.

<15:08> REBOOT CODE
80 hex

<07:00> REBOOT ADDRESS
Data to console address (33 hex) used to force the reboot to occur.



The contents of this register are generated by the console during Control Store and Dispatch RAM ECC correction. The console sends this register to the CPU as part of the ECC correction process.

<31> CORRECTION FAILURE
Indicates that the console was unable to correct the Control Store or Dispatch RAM identified in the CS/DRAM ID (<31:30> <02:00>).

<30:29> RESERVED

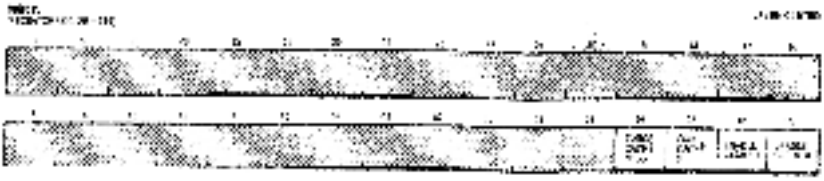
<28:16> CONTROL STORE ADDRESS
Address the console used when correcting the Control Store or DRAM Parity Error.

<15:08> CONTROL STORE SYNDROME
This is the syndrome the console used to correct the bit in error.

<07:03> RESERVED

<02:00> CONTROL STORE/DRAM IDENTIFICATION CODE
This is the code passed to the console from the EBox to determine what Control Store to apply correction to.

CODE	MEANING
0	No Error
1	EBox Control Store Parity Error
2	IBox Control Store Parity Error
3	IBox DRAM Parity Error
4	EBox DRAM Parity Error
5	FBox Adder Module Control Store Parity Error
6	FBox Multiplier Module Control Store Parity Error
7	MBox Control Store Errors (Data, Address, or Micro-stack)



This register is made up of MBOX Register 04 (byte 0).

<31:08> RESERVED
<07:00> SOURCE: MAPP (REG) = MBOX REG 04 (ADDR CONTROL 1)

<07:04> RESERVED

<03> FORCE CACHE MISS
REG4 FORCE CACHE MISS (MAPP)
When set, forces a cache miss (overrides CACHE 1 ENABLE and CACHE 0 ENABLE). This bit is used for forcing a cache refill during diagnostics.

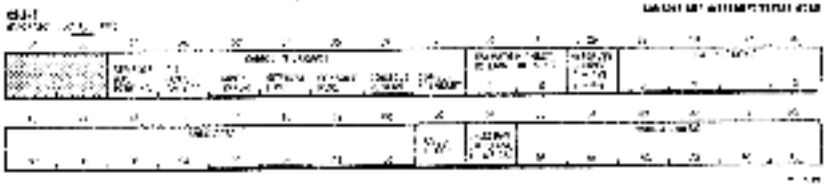
<02> FORCE CACHE HIT
REG4 FORCE CYC (MAPP)
This bit works in conjunction with CACHE 1 (MAPP) and CACHE 0 ENABLE as shown in the following table:

Force Hit	C1 Enable	C0 Enable	Function
0	0	0	Cache Miss
0	0	1	Cache 1 determines Hit or Miss
0	1	0	Cache 1 determines Hit or Miss
0	1	1	Both Caches determine Hit or Miss
1	0	0	Cache Miss
1	0	1	Force Hit in Cache 0
1	1	0	Force Hit in Cache 1
1	1	1	Illegal, gives bad parity error with Write Bit = 1

The three bits <02:00> reflect the following:
Code 000 - Cache 0 and 1 Disabled
Code 001 - Cache 1 Disabled
Code 010 - Cache 0 Disabled
Code 011 - Normal Running Code.
Code 100 - Cache 0 and 1 Disabled
Code 101 - Used to force a sweep of Cache 0 if Written Bit is set. Also used for diagnostic purposes to force a hit in a given cache.
Code 110 - Used to force a sweep of Cache 1 if Written Bit is set. Also used for diagnostic purposes to force a hit in a given cache.
Code 111 - This is an illegal code that forces the case of a hit in both caches. A bad parity error with WR1 is also formed if, during normal operation, both caches give indications of a hit. This is considered an MBOX FATAL ERROR.

<01> ENABLE CACHE 1
REG4 CACH 1 ON (MAPP)
When set, enables Cache 1 (allows Cache 1 to be read and written).

<00> ENABLE CACHE 0
REG4 CACH 0 ON (MAPP)
When set enables Cache 0 (allows Cache 0 to be read and written).



CSLINT (Console and Interrupt Register)

This is a two port register. Bits <31:16> control the System Interrupt Enables and bits <15:00> reflect the CBox Status.

<31:30> RESERVED

<29:23> CPU INTERRUPT
Setting a bit in this field will result in a CPU interrupt.

<28> CONSOLE HALT PENDING
CBOX CTL RD
Posted by the console, indicates that the console wants to interrupt the CBox and force it to execute a Console Halt. The CBox micro code will be forced to the CON wait loop.

<29> CPU POWER FAIL
EBOX CPU PF INTR LVL3
Posted by the console. Indicates that the EBOX has not (and the console expect it) invoking power failure and has approximately 10 ms to shut down in an orderly manner.

<27> MBOX ERROR
EBOX MBOX INTR LVL3
Posted by the MBox. Indicates that the MBox detected an error on any type (excluding IB parity errors). The interrupt is handled by the EBox at IPL 15 (hex) <15>.

<26> INTERVAL TIMER
EBOX TIME INTR LVL3
Posted by the CBox. Indicates that the Interval Count Register has overflowed.

<25> CONSOLE BLK2
EBOX RL INTR LVL3
Posted by the console. Indicates that the console wants to interrupt the MBox to transfer a byte to or from the BLK2.

During an BLK2 read (SNAP files, etc.), the PII assembles 512 bytes and then transfers them to the MBox a byte at a time via this interrupt mechanism.

During an BLK2 write, the PII interrupts the EBox for each 32 data unit it has assembled 512 bytes. It then transfers the data to the BLK2.

<24> CONSOLE RECEIVE
EBC2 TRX INTR LVL3
Posted by the console. Indicates that the console has acknowledged the last CPU transmission.

<23> CONSOLE TRANSMIT
EBC2 TTX INTR LVL3
Posted by the console. Indicates that the console wants to transmit a byte of information to the CPU via the CBus.

<22:21> IOA WITH HIGHEST INTERRUPT REQUEST
EBC1 EXT INTR SRC <1:0>
This field indicates the I/O Adapter (SBIA) that has the highest IPR.

<20> INTERRUPT SOURCE (0=EXT/1=INT)
EBC3 INT INTR
Set by hardware. When set, indicates that the source of the pending interrupt is internal. When clear, indicates that the source of the pending request is external.

<19:16> EBOX IPR STATUS <3:0>
EBC3 ACTIVE IPR 3:0
This field represents the least significant four bits of the highest active hardware (internal or external) interrupt request during the previous machine cycle. When this field is zero, there was no hardware interrupt.

<15:08> CONS DATA <07:00>
EBC2 CBUS IN REG 07:00
Although the register is used primarily to receive data from the console, all data transfers between the console to the CPU are latched in this register.

<07> CBUS CLOCK
EBC2 CBUS CLOCK
This bit represents the state of the CBus Clock line. The CBus Clock must be asserted by writing a one to this bit and negated in a subsequent microinstruction by writing a zero to this bit in order to complete a CBus read or write operation.

<06> CBUS R/W (0=RD (ST/1=WR (SL)
EBC2 CBUS WRITE
This bit determines whether a CPU read or write operation is to be performed over the CBus. The EBox is enabled to drive the CBus data lines when this bit is set. The console is enabled to drive the CBus data lines when this bit is reset.

<05:00> CBUS ADDRESS <A5:A0>
EBC2 CBUS A5:A0
The address of the console location to be read or written is loaded into this register.



CSM.STATUS (CSM Status Register)

This is the status word used to control the Console Support Microcode (CSM). It is located in SSC C8.

<31:16> DOUBLE ERROR STATUS
This field is normally equal to zero. If this field is non-zero then it indicates that a non-CMOS double error condition has occurred.

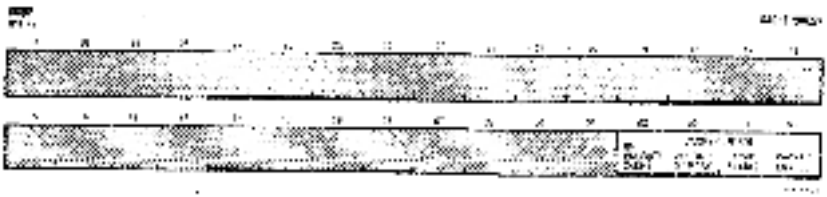
<15:08> EXECUTION STATUS

If this field is equal to zero it indicates that CSM is executing in the EBox. If this field is not equal to zero (i.e., non-zero) then it indicates that the VAX ISP microcode is executing in the EBox.

<07:00> ENTRY CODE
This field contains a code that indicates reason why CSM was started.

Code Reason

00 CSM is not executing.
04 Interrupt Stack not Valid (Software Error).
05 A non-EBox double error was detected.
06 Halt Instruction was decoded in Kernel Mode.
07 SCB Vector Code <1:0> = 3 (Software Error).
08 SCB Vector Code <1:0> = 2 (no MCS Microcode).
09 Pending Error on Halt.
0A CSMx Termination Opcodes and Interrupt Stack = 1.
0B CSMx Instruction decoded and Vector <1:0> / 0.
10 Micro break was encountered which caused the console to start CSM at CSM.ENTRY.MICRO:.
11 Console Halt Pending was set which caused CSM to start running on ARark at A.CSM.ENTRY.MICRO:.
15 The CPU was powered up and the Console started CSM at CSM.INIT.POR:.
16 During the power up sequence this code is used by the FIND 64KB and FIND 8KB procedures to interface with CSM.ERROR: routine. The Console should never see this code. If it does then there something very wrong in the CPU.



<31:04> HIGH WORD

<03:00> CACHE CONTROL

<03> INVALIDATE
When set, clears written and valid bits and does not update memory. Both caches are always invalidated, regardless of whether they were previously enabled. The cache access code is then set to be write-through or both cache halves as expressed by C0 ENA and C1 ENA.

State of Cache after Operation:

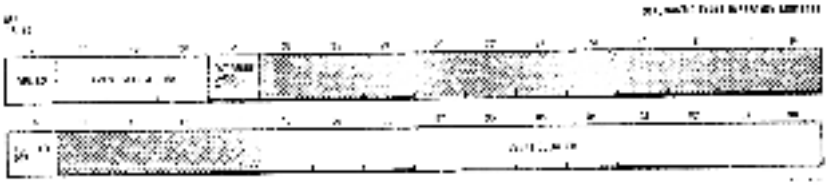
CACHE 1 ENABLE	CACHE 0 ENABLE	
0	0	Both Caches are off
0	1	Cache 0 is active Cache 1 is off
1	0	Cache 1 is active Cache 0 is off
1	1	Cache 0 and 1 are active

<02> VALIDATE MEMORY
When set, copies cache written locations to memory, according to the previous enable bits. The cache access code is then set to select cache halves as expressed by C0 ENA and C1 ENA in the table above.

If neither INV or VAL are set, the only effect of an MTPR CADR will be to enable the cache according to bits <01:00>.

<01> CACHE 1 ENABLE

<00> CACHE 0 ENABLE



<31> ARMED
When set, fault insertion is armed and controlled by the hardware associated with this register.

<30:28> EVENT SELECT CONTROL<2:0>
These bits form the count modes for the EVENT COUNTER increment encode. When enabled, the counter increments at CLK TSC.

The following count modes exist:

VAL [2:0]	MODE
0	NEVER COUNT
1	ALWAYS COUNT
2	-ISTALL
3	-ISTALL
4	IBOX UMARK + IBOX UMARK
5	<IBOX CMARK * -ISTALL> + <IBOX UMARK * -ISTALL>
6	IRD * -ISTALL

<27> OVERRIDE MICRO CHMK MARK
When set, overrides microcode mark as one of the conditions needed to insert the fault. In other words, fault insertion occurs when the EVENT COUNTER overflows.

<26:16> RESERVED

<15> IP LOAD PTR
When set, causes PTR in CI to be loaded into PR location corresponding to virtual address in RC. All other bits are ignored. This bit does not exist in hardware. It is recognized by MTPR DFT microcode (Maxon condition), and may be used by macrocode to generate IP parity errors.

<14:11> RESERVED

<10:00> EVENT COUNTER
The event counter is capable of counting 1 to 2047 cycles, IBox and IBox microcode mark bits with and without stalls, unstalled IBox cycles, unstalled IBox or Mbox cycles, or an external event until EVENT COUNTER OVERFLOW occurs.



<31>	PARITY FAULT

<30>	WRITE SEQUENCE FAULT

<29>	UNEXPECTED READ DATA

<28>	RESERVED

<27>	MULTIPLE TRANSMITTER FAULT

<26>	TRANSMITTER TIMING FAULT

<25>	RESERVED

<24>	EXTERNAL ABORT

<23>	POWER DOWN

<22>	POWER UP

<21>	RESERVED

<20>	INTERRUPT ENABLE

<19>	PACKET INTERRUPT

<18>	ABORT

<17>	HALT

<16>	UNSOLICITED READ DATA

<15>	READ DATA SUBSTITUTE

<14>	COMMAND/ADDRESS TIME OUT ID1

<13>	READ DATA EXPECTED TIME OUT ID1

<12>	RECEIVED ERROR CONFIRMATION ID1

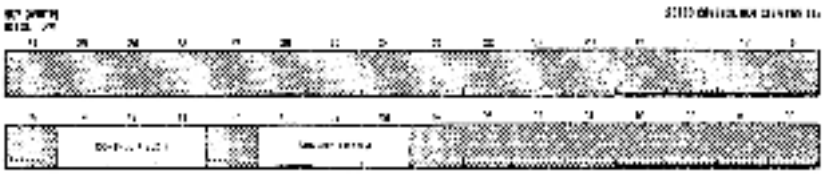
<11>	DATA INTERCONNECT STALL

<10>	COMMAND/ADDRESS TIME OUT ID2

<09>	READ DATA EXPECTED TIME OUT ID2

<08>	RECEIVED ERROR CONFIRMATION ID2

<07:05>	ADAPTER TYPE CODE
Adapter type code = 101(B)



<3:15> RESERVED

<14:12> CONTROL FIELD B

14	13	12	
0	0	0	No operation
0	0	1	Clear corrected read data (DCR read <16>)
0	1	0	Set external abort (DCR read <24>)
0	1	1	Clear backup interrupt (DCR read <19>)
1	0	0	Reset
1	0	1	Set out of sequence test
1	1	0	Clear out of sequence test
1	1	1	No operation

<11> RESERVED

<10:08> CONTROL FIELD A

10	09	08	
0	0	0	No operation
0	0	1	Clear power up (DCR read <22>)
0	1	0	Clear power down (DCR read <23>)
0	1	1	No operation
1	0	0	Clear abort interrupt and read data substitute (DCR read <1:0, 16>)
1	0	1	Clear interrupt enable
1	1	0	Set interrupt enable
1	1	1	Clear halt

<07:00> RESERVED



<31:30> RESERVED

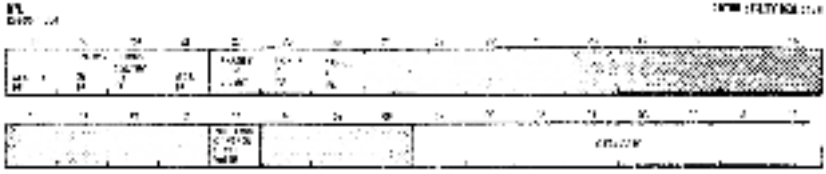
<29:17> DA780 ADDRESS SPACE
These bits are used as the ADDRESS SPACE available to the DA780. The addresses for the DA780 fall into the adapter address space as do all hardware addresses. Bit 29 is set to 1, and <25:26> and <24:17> are all 0's. Bit 25 is 0 for SBIA 0 and 1 for SBIA 1.

<16:13> TRANSFER REQUEST IDENTIFIER
The number for the transfer request bits is set when the DA780 is installed. The number used in this field can be any numbers from 1 through 15.

<12:11> MBZ
Must be zero.

<10:09> PAGE SELECT
These bits inform the CPU which page in the DA780 is to be accessed. This register contains a table of page selection bits.

<08:00> SBI ISTAR ADDRESS



<3:> GENERAL PARITY ERROR
Indicates that a parity error was detected either on the CI, the DI, or the WCS. This bit is generated on the WCS Interface Module.

<30> DEVICE INTERCONNECT PARITY ERROR
Indicates that the DGA detected a parity error on the DI. This bit is generated on the SIU Module.

<29> CONTROL INTERCONNECT PARITY ERROR
Indicates that the Microprocessor Module detected a parity error on the CI. This bit is generated on the DCP.

<28> WRITABLE CONTROL STORE PARITY ERROR
Indicates that the DGA detected a Writable Control Store RAM Parity Error. This bit is generated on the Microprocessor Module.

<2> ENABLE DEVICE INTERCONNECT PARITY ERROR ABORT
When this bit is set and a DI parity error is detected it will cause the DR780 to abort the data transfer and notify the processor.

<26> FORCE DEVICE INTERCONNECT PARITY ERROR
This is a diagnostic feature used to test the DI parity circuits on the DGA. When set this bit will force a parity error on the DI.

<25> FORCE CONTROL INTERCONNECT PARITY ERROR
This is a diagnostic feature used to test the CI parity circuits on the DCP. When set this bit will force a parity error on the CI.

<24:12> UNDEFINED

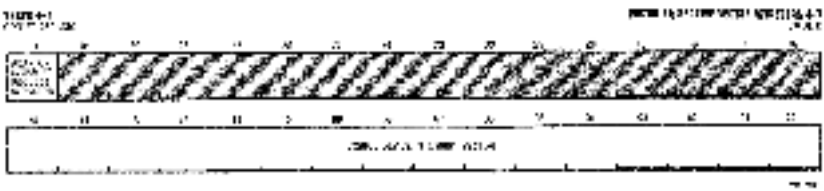
<1> WRITABLE CONTROL STORE VALID
When set this bit allows the DR780 Microprocessor to operate. When cleared the DR780 aborts the data transfer in progress.

<16:08> UNDEFINED

<07:00> DATA RATE
Specifies the 2's complement of the DI data transfer rate in megabytes/second. The data transfer rate (i.e. MB/sec.) is calculated using the following formula: DR = 40/256. Where DR is the value of the data rate in bytes.

NOTE

The maximum data rate is five (eight MB/sec). Any rate less than five prevents this field from being loaded and the field will retain the previous value.



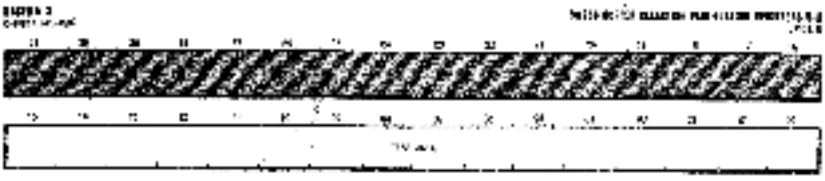
-A BERVR VECTOR REGISTERS 4-7

Note: The UBA contains four BERVR's: BRRVR 7, BRRVR 6, BRRVR 5, BRRVR 4. Each BRRVR corresponds to a Unibus interrupt bus request level: 7, 6, 5, or 4. Each BRRVR is a read only register and will contain the interrupt vector of the Unibus device interrupting at the corresponding BR level. Each BRRVR is read by the software as a part of the UBA interrupt service routine. Note that the UBA interrupt service routine is the routine to which the VAX 8600/8650 CPU will transfer control once it has determined that the UBA or the Unibus has issued an interrupt request to the CPU.

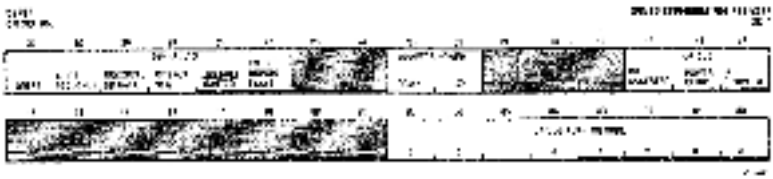
<31> ADAPTOR INTERRUPT REQUEST INDICATOR
0 = No UBA interrupt pending
1 = UBA interrupt pending

<10:16> RESERVED

<9:00> DEVICE INTERRUPT VECTOR FIELD
These bits contain the device interrupt vector loaded by the UBA during the Unibus interrupt transaction.



<15:00> PALMD INTRQ ADDRESS BITS <17:02>
BUFFER SELECTION VERIFICATION REGISTERS 0-3
Note: These four read/write diagnostic registers are provided to give the diagnostic software a means of accessing and testing the integrity of the data path BAN. Four locations in the data path BAN have been assigned to these registers. Writing and reading the D08VR's has no effect on the behavior of the BAN.



<31:27> SBI FAULTS
These bits are set when the UBA detects specific fault conditions on the SBI. Note: These bits cannot be set once FAULT has been asserted. The negation of FAULT and the clearing of the fault conditions clear the bits. The setting of any bits <31:27> will cause the UBA to assert the FAULT signal for one cycle on the SBI during the confirmation time for the associated transfer.

<31> PARITY FAULT
This bit is set when the UBA detects an SBI parity error.

<30> WRITE SEQUENCE FAULT
This bit is set when the UBA receives a write masked or interlock write masked command and does not receive the expected write data in the following cycle.

<29> UNEXPECTED READ DATA FAULT
This bit is set when the UBA receives read data when a read command was not issued.

<28> INTERLOCK SEQUENCE FAULT
This bit is set when an interlock write masked command to the Unibus address space is received by the UBA without a previous interlock read mask command.

<27> MULTIPLE TRANSMITTER FAULT
This bit is set when the UBA is transmitting on the SBI and the ID bits received do not match the ID it transmitted.

<26> TRANSMIT FAULT
This bit is set if the UBA was transmitting during a detected fault condition. Note: When the software subsequently reads the configuration and status registers of each of the nexus on the SBI in order to identify the source of the fault, the UBA will be identified as that source if bit 26 is set.

<25:24> RESERVED

<23> ADAPTOR POWER DOWN
Set when the UBA power supply asserts AC LO. Cleared by writing a 1 to this bit or when Adaptor Power Up is set.

<22> ADAPTOR POWER UP
Set by negation of AC LO. Cleared by writing a 1 to this bit or when Adaptor Power Down is set.

<21:19> RESERVED



<10> UNIBUS INITIALIZATION ASSERTED (UBINIT)
This bit is set by the execution of Unibus Init. It is cleared by setting the Unibus Initialization Complete bit (UBIC) or by writing a 1 to this bit.

<17> UNIBUS POWER DOWN (UBPDN)
Set when Unibus AC LO is asserted. Notes: This indicates that the Unibus has initiated a power down sequence. The setting of the UBIC bit or writing a 1 to this bit will clear UBPDN.

<16> UNIBUS INITIALIZATION COMPLETE (UBIC)
This bit is set by a successful completion of a power up sequence on the Unibus. Note: It is the last of the status bits that can be set during a UBA initialization sequence, and it can be interpreted to mean that the UBA and the Unibus are ready. The assertion of Unibus AC LO or Unibus Init, or the writing of a 1 to this bit location clears this.

<07:00> ADAPTOR CODE
These bits define the node assigned to the UBA.

ADAPTOR CODE BIT ASSIGNMENT

Bit Number	7	6	5	4	3	2	1	0
Value	0	0	1	0	1	0	Vb	Va
UBA Number							Vb	Va
0							0	0
1							0	1
2							1	0
3							1	1



<31> SPARE
This read/write bit has no affect on any UBA operation. Note: SBI DEAD, Adaptor INIT, and a power up sequence on the UBA will clear this bit.

<30> DISABLE INTERRUPT
When set, this bit will prevent the UBA from recognizing interrupts on the Unibus. It is useful in testing the response of the UBA to the passive release condition during a Unibus interrupt transaction. SBI DEAD, Adaptor INIT, and the power up sequence on the UBA will clear this bit.

<29> DISABLE MAP PARITY
When set, this read/write bit will inhibit the parity bits of the map registers from entering the map registers' parity checkers. Note: The map register parity generator/checkers generate and check parity on eight bit quantities. Each parity field (eight data bits and one parity bit) is implemented so that the total number of 1's in the field is odd.

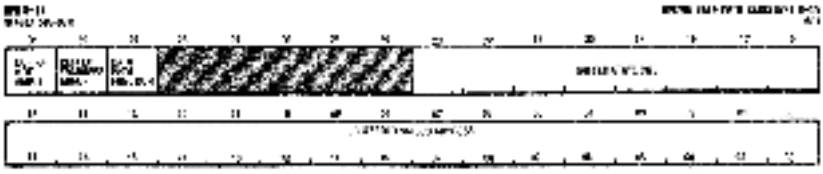
<28> DISABLE DATA PATH PARITY
When set, the DDPP bit will inhibit the parity bits of the data path RAM from entering the parity checkers. Note: The DDPP bit functions in the same manner as the DMP bit. The data path parity generator/checkers generate and check parity on eight bit data units. Each parity field (eight data bits and one parity bit) is implemented so that the total number of 1's in the field is odd. When the integrity of the parity generator/checkers is to be tested through use of the DDPP bit, the total number of 1's in at least one of the bytes of data must be even. With the parity bit disabled by the DDPP bit, a data path parity failure will result during a Unibus to MBP read, a MBP to SBI write, or a purge. SBI DEAD, Adaptor INIT, and the power up sequence on the UBA will clear the DDPP bit.

<27> MICROSEQUENCER OK
The MIC OK bit is a read only bit which indicates that the UBA microsequencer is in the idle state. The microsequencer will enter the idle state after it has completed the initialization sequence or once it has completed a UBA function.

The MIC OK bit can be used by the diagnostic to determine whether or not the microsequencer has completed a successful power up sequence and whether or not it is caught up in any loops. Make that SBI DEAD, UBA power supply DC LO, and Adaptor INIT force the microsequencer into the initialization routine. Once the routine has been completed and the microsequencer has entered the idle state, MIC OK will be true (1).

<26:24> RESERVED

<23:00> These bits are the same as bits <23:00> of the DW780 configuration register.



<31> BUFFER NOT EMPTY
Note: Each DB3 contains a data path status bit called buffer not empty.

1 = Buffer not empty
0 = Buffer empty

The BNE bit reflects the state of the associated BDP. If this bit is set (1), the BDP contains valid data. If clear, then the BDP does not contain valid data. The CPA uses the bit to determine the proper action for DMA transfers via the BDP. If bit 31 is set as a DATI transfer begins, the data in the BDP will be asserted on the Unibus. If bit 31 is clear on a DATI, the DWA will initiate a read transfer to SBI memory, gate the addressed data to the Unibus, and then load the read data into the BDP, thereby setting bit 31.

For a DMA write transfer via the associated BDP, the BNE bit is set each time Unibus data is loaded into the BDP. The bit is then cleared when the contents of the BDP are transferred to SBI memory.

The software will write a 1 to the BDE bit to initiate a purge operation at the completion of a DMA transfer using the corresponding buffered data path. The DWA executes purge operations as follows:

1. WRITE TRANSFERS TO MEMORY
This bit is set if any bytes of data remain in the corresponding BDP. The bit is cleared if no data remains to be transferred. Note: The DWA will transfer this data to the SBI location addressed. The DWA will then initialize the BDP and clear the BNE bit. The purge operation will be treated as a no-op (it is a legal do-nothing function).

2. READ TRANSFERS TO MEMORY
Note: If any bytes of data remain in the BDP, the DWA will initialize the BDP by clearing the BNE bit.

In addition, the following considerations apply to the purge operation:

- For purge operations in which data is transferred to memory, the SBI transfer takes about 2 us. The DWA will not respond to data path register read transfers during this period (busy confirmation), thereby preventing a race condition when testing for BNE bit.
- A purge operation to data path register 0 (direct data path) is treated by the DWA as a no-op.

<30> BUFFER TRANSFER ERROR
Note: This is a read-write-one-to-clear bit. The DMA sets the BTE bit if a failure occurs during a BDP to SBI write or a purge, or for a buffer parity failure during a Unibus to BDP read access. If bit 30 is set, any additional DMA transfers via the BDP will be aborted until the bit is cleared by the software. Note that if a parity error on the Unibus occurs during a DMA read, the Unibus signal PE will be asserted, giving the Unibus device the opportunity to abort its own DMA transfer. If the device does not abort its own transfer, the DMA will abort the transfer on the next access. The purge operation does not clear the BTE bit. The software clears this bit by writing a 1 to bit location.

<29> DATA PATH FUNCTION
Note: The BDP is a read only bit. This bit indicates the function of the DMA transfer using this data path.

0 = DMA read
1 = DMA write

<23:16> BUFFER STATE
Note: These eight read only bits indicate the state of each of the eight byte buffers of the associated BDP during a DMA write transfer. They are included in the data path register for diagnostic purposes only. The CBA generates the SBI mask bits from the MS bits during a BDP to SBI write transfer or purge operation. The bits are set as each byte is written from the Unibus. The bits are cleared during the SBI write operation.

0 = Empty
1 = Full

<15:00> BUFFERED UNIBUS ADDRESS
Note: This portion of each DPR contains the upper 16 bits of the Unibus address, DA <17:02>, generated during a Unibus to BDP write transfer using the associated BDP. If the transfer through the associated BDP is in the byte offset mode, and the last Unibus transfer has spilled over into the next quadword, then these bits contain DA <17:02> + 1. Notation:

BUBA <15:00> = Upper 16 bits of DA <17:00> + Byte Offset

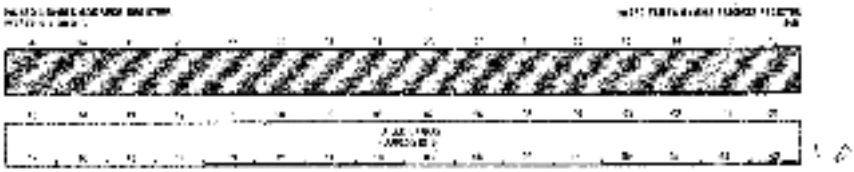
Note: This is the Unibus address from which the SBI address will be mapped during a purge operation.



<31:08> RESERVED

<07:00> MAP REGISTER NUMBER
These bits contain the binary value of the number of the map register that was in use at the time of a failure. Bits <08:00> correspond to bits <17:09> of the Unibus address.

Note: The Failed Map Entry Register contains the map register number used for either a DMA transfer or a purge operation that has resulted in the setting of one of the following status register error bits: IVNR, NPGF, DPPE, CXING, CXIBR, RDS, NIUU. This register is locked and unlocked with the Unibus to SBI data transfer error field of the status register. The EMER is a read only register. Attempts to write to the EMER will result in an SBI error confirmation. When the EMER is not locked, its contents are invalid.

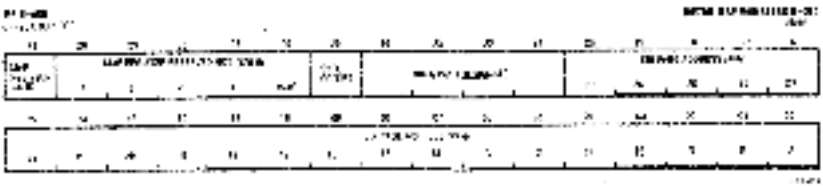


<31:18> RESERVED

<17:02> FAILED UNIBUS ADDRESS BITS <17:02>

Note: The FAILED UNIBUS ADDRESS REGISTER contains the upper 16 bits of the Unibus address translated from an SBI address during a previous software initiated data interface. The occurrence of either of two errors indicated in the status register will lock the FAILED UNIBUS ADDRESS REGISTER: Unibus Select Time Out (UBSTO) and Unibus Slave Sync Time Out (NXSSTMO). When the error is cleared, the register will be unlocked.

The FAILED UNIBUS ADDRESS REGISTER is a read only register. Attempting to write to the register will result in an error confirmation. No signals or conditions will clear the register. The contents of the FAILED UNIBUS ADDRESS REGISTER are listed below.



<31> MAP REGISTER VALID (MRV)
The MRV bit is set by software to indicate that the contents of this map register is valid. The MRV is tested each time a map register is accessed. If the bit is set, the transfer continues. If the bit is not set, the Unibus transfer is aborted (MXF) and the invalid map register bit (IVMR) is set in the UBA status register, provided that the map register has not been disabled by the map register disable (MRD) bits of the control register.

<30:27> RESERVED

<26> LONGWORD ACCESS ENABLE (LWAE)
If set, and the map register selects a buffered data path (BDP), then the longword aligned 32-bit random access mode is enable for the BDP. This bit has no effect if the direct data path is selected by the map register. This bit is cleared on initialization.

<25> BYTE OFFSET BIT
This read/write bit is set if "this Unibus page" is using one of the BDP's, and the transfer is to an SBI memory address. This bit is cleared by initialization. Note: When the UBA will perform a byte offset operation on the current Unibus data transfer. The software can interpret this operation as increasing the physical SBI memory address, mapped from the Unibus address, by one byte. This allows word-aligned Unibus devices to transfer to odd byte memory addresses.

Unibus transfers via the DOE or to SBI I/O addresses will ignore the byte offset bit.

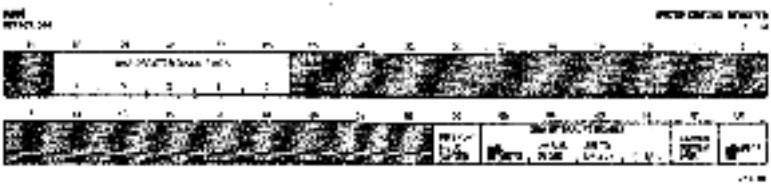
<24:21> DATA PATH DESIGNATOR BITS

Notes: 0000 = Direct Data Path (DDP)
0001 = Buffered Data Path 1
1111 = Buffered Data Path 15

The data path designator bits are read/write bits that are set and cleared by the software to designate the data path that "this Unibus page" will be using.

The software can assign more than one Unibus transfer to the BDP. The software must ensure that not more than one active Unibus transfer is assigned to any BDP.

<20:09> SBI PAGE ADDRESS
The "Unibus page" will be mapped. These bits perform the Unibus to SBI page address translation. When an SBI transfer is initiated the contents of SPA <27:09> are concatenated with Unibus address bits UA <08:00> to form the 28 bit SBI address.



<31> RESERVED

<30:36> MAP REGISTER DISABLE [MRD]
This read/write field disables Map Registers in groups of sixteen according to the binary value contained in the field. The MRD bits prevent the UBA from responding to a Unibus address that points to a disabled Map Register. The software will load this field with a binary value equal to the number of 4K word units of memory attached to the Unibus as follows:

MAP REGISTER DISABLE BITS

MRD <4:0>	AMOUNT OF UNIBUS MEMORY (WORDS)	MAP REGISTERS DISABLED
00000	0K	NONE
00001	4K	0 - 15 (10)
00010	8K	0 - 31 (10)
00011	12K	0 - 47 (10)
.	.	.
.	.	.
.	.	.
11110	120K	0 - 480 (10)
11111	124K	0 - 495 (10) (ALL)

<25:07> RESERVED

<06> INTERRUPT FIELD SWITCH [IFS]
This bit determines whether interrupts from a Unibus-Out side of the UBA will be fielded by the VAX8xxx CPU or passed to the Unibus-In side of the UBA. If this bit is set and if bit <05> (BRIE) is set, then the interrupt will be passed to the CPU. If this bit is cleared, then the interrupt will be passed to the Unibus-In side of the UBA and it is not seen by the SBI.

<05> BUS REQUEST INTERRUPT ENABLE [BRIE]
When this bit is set, the UBA is allowed to pass interrupts from the Unibus to the CPU, providing that bit <06> [IFS] is set. It is cleared by the Adapter INIT, SBI UNIAK, and SBI CNFG signals. Note: The power up state of the BRIE bit is 0.

<04> UNIBUS TO SBI ERROR FIELD INTERRUPT ENABLE (USEFIE)
This bit enables an interrupt request to the CPU whenever any of the following DW750 Status Register bits are set during a DMA transfer.

<10> - RDTO (Read Data Time Out)
<09> - RDS (Read Data Substitute)
<08> - CXTER (Command Transmit Error)
<07> - CXTMO (Command Transmit Time Out)
<06> - DPPE (Data Path Parity Error)
<05> - IVMR (Invalid Map Register)
<04> - MRPE (Map Register Parity Failure)

The power up state of the USEFIE bit is 0. SBI UNJAM and Adaptor INIT will clear USEFIE.

<01> SBI TO UNIBUS ERROR FIELD INTERRUPT ENABLE
If this bit is set, the UBA will generate interrupt requests to the CPU when either of the following Ja709 Status register bits are set.

<01> - UBSTO (Unibus Select Time Out)
<00> - UBSSYNTO (Unibus Slave Sync Time Out)

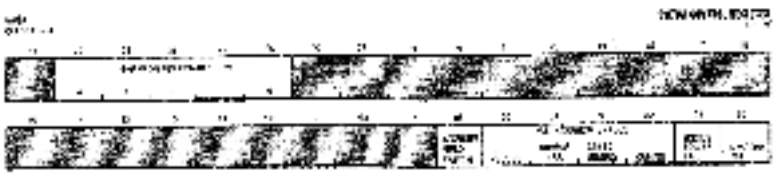
This bit is cleared when in the power up state. SBI UNJAM, SBI DEAD and Adapter INIT will clear the USEFIE bit.

<02> CONFIGURATION INTERRUPT ENABLE
If this bit is set, the UBA will initiate an interrupt request to the CPU whenever any of the following Environmental Status bits in the Configuration Register are set.

<19> - AD PUN (Adaptor Power Down)
<18> - AD PUP (Adapter Power Up)
<17> - UB INIT (Unibus Init Asserted)
<16> - UB PUN (Unibus Power Down)
<15> - CBIC (Unibus Initialization Complete)

This bit is set when the power up state is set. It is cleared by Adapter INIT, SBI UNJAM and SBI DEAD.

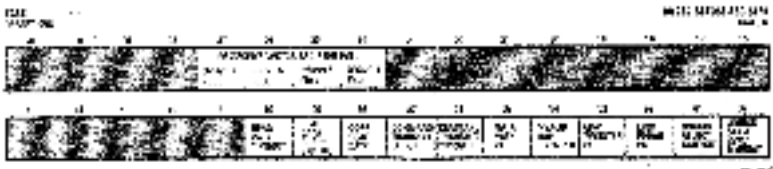
<01> UNIBUS POWER FAIL
This bit is set when a power fail sequence on the Unibus, asserting AC LO, DC LO and INIT are in their correct sequence. The software uses this bit to initialize the Unibus. The Unibus will remain powered down as long as DPF is set. It is cleared when a Unibus power up sequence or when the Unibus power down sequence has finished and Unibus power is OK.



<OU> ADAPTER INIT
When this bit is set it will completely initialize the UBA and the Unibus. The Map Registers, the Data Path Registers, the Status Register and the Control Register will be cleared. The UBA will start the initialization routine in the microsequencer, and it will generate a DC LOW initialization sequence and a Unibus power fail sequence on the Unibus. The UBA initialization sequence takes only 700 ns to complete, while the Unibus power fail sequence requires approximately 25 ms.

Only the Configuration Register and the Diagnostic Control Register can be read during the adapter initialization sequence. Only the Configuration Register, the Diagnostic Control Register, the Control Register and the Status Register can be written during the adapter initialization sequence.

Once the sequence has been completed, all UBA registers can be accessed. However, the Unibus cannot be accessed until the Unibus initialization sequence has been completed as well. The software can look for this condition by reading bit <16> (UPIC) of the Configuration Register, or by waiting bit <02> (INTFIN) of the Control Register and looking for the interrupt generated by the setting of the UPIC bit. Note, however, that the assertion of either Unibus INIT or Unibus power down will also initiate an interrupt (UBINIT). The Adapter Init bit can be set by writing a 1 to the bit location; it is self-clearing.



<31:28> RESERVED

<27:24> BR RECEIVE VECTOR REGISTER FULL
These bits indicate the state of the SBI addressable BRRVR's. Each bit is set when the interrupt vector is loaded into the corresponding BRRVR during a Unibus interrupt transaction, providing that the SBI processor is fielding Unibus device interrupts.

Each bit is cleared by the successful completion of a read data transaction following a read BRRVR command. The software will use these bits set only after a read data failure has occurred during the execution of a read BRRVR command and the Unibus interrupt vector has been saved by the UBA. These bits are cleared only by a subsequent read to the corresponding BRRVR or by an adaptor initialization sequence.

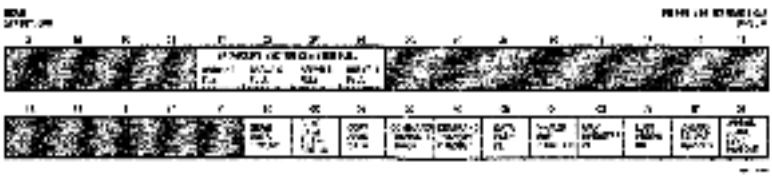
Bit 27 = BRRVR 7 Full
Bit 26 = BRRVR 6 Full
Bit 25 = BRRVR 5 Full
Bit 24 = BRRVR 4 Full

<23:11> RESERVED

<10> READ DATA TIME OUT (RDTO)
The UBA sets the RDTO bit when the following conditions are all true. A Unibus device has initiated a DMA read transfer. The UBA has successfully transmitted a read command on the SBI. The SBI memory or UBIA has not returned the requested data within 100 us, and the Unibus device has not timed out. Note that the normal Unibus time out is 10 us, and that after 10 us, the Unibus device will set its non-existent memory bit. Thus, the RDTO bit will be set on the UBA status register only if the Unibus device timeout function is inoperative, or takes more than 100 us. This bit is not set for a RDP to SBI protocol. Also see Note 1.

<09> READ DATA SUBSTITUTE (RDS)
This bit is set if a read data substitute is received in response to a Unibus to SBI read command (DMA read transfer). No data will be sent to the Unibus device, and when the device timeout occurs it will set the non-existent memory bit on the device. Also see Note 1.

<08> CORRECTED READ DATA (CRD)
The UBA sets this bit when it receives corrected read data in response to an SBI read command during a DMA read transfer.



<07> COMMAND TRANSMIT ERROR (CXTER)
This bit is set when the UBA receives an error confirmation in response to an SBI command transmission during a Unibus to SBI access, a BDP to SBI read, a BDP to SBI write, or a purge operation. This bit is not set for a BDP to SBI prefetch. Also see Note 1.

<06> COMMAND TRANSMIT TIMEOUT (CXTMO)
This bit is set when the UBA fails to complete an SBI command transfer within 100 nA for any of the following operations:

1. A BDP to SBI write
2. A BDP purge operation
3. A BDP to SBI read operation for which the Unibus device has timed out

This bit is not set for a timeout for a BDP to SBI prefetch. Also see Note 1.

<05> DATA PATH PARITY ERROR (DPPE)
This bit is set when a parity error in a buffered data path occurs during either a Unibus to BDP read, BDP to SBI write, or a BDP purge operation. Also see Note 1.

<04> INVALID MAP REGISTER (IVMR)
The UBA sets this bit during a Unibus DMA transfer or purge operation when the Unibus address points to a map register that has not been validated by the software and has not been disabled by the NXD bits. Also see Note 1.

<03> MAP REGISTER PARITY FAILURE (MRPF)
This bit is set with the occurrence of a map register parity error during one of the following operations:

1. A Unibus access in which the Unibus address points to a map register that has a parity error in the upper 14 bits, providing that the map register has not been disabled by the NXD bits.
2. Mapping a Unibus address to an SBI address during a direct data path to SBI operation or a BDP to SBI read operation (but not during a prefetch).
3. Mapping an address from a buffered data path to an SBI address during a purge operation or a BDP to SBI write. Also see Note 1.

<02> LOST ERROR BIT
The UBA sets this bit if the locking error field is locked and another error within this field occurs. The lost error bit does not initiate an interrupt request. See Note 1.

<01> UNIBUS SELECT TIME OUT (UBSTO)
The UBA sets this bit if it cannot gain access to the Unibus within 50 us in the execution of a software initiated transfer (SBI to Unibus transfer). Note: When UBSTO is set it indicates that the UBA has issued BBSY on the Unibus but has not become bus master. This condition indicates the presence of a hardware problem on the Unibus. The Unibus may be inoperative, or one device may be holding it for extended periods. Note that if the Unibus does become inoperative, it may be possible to clear the problem with the assertion of UBInit on the SBI, the setting and clearing of the Unibus Power Fail bit (control register bit 1) or the setting of Adapter INIT (control register bit 0).

<00> UNIBUS SLAVE SYNC TIME OUT (UBSSYNTO)
This bit is set when a SBI to Unibus transfer (software initiated transfer) times out during the data transfer cycle on the Unibus. The timeout occurs after 12.8 us. This bit indicates a transfer failure resulted when a Non-Existent Memory or device on the UNIBUS is addressed. The above two bits, UBSTO and UBSSYNTO, form an SBI to Unibus transfer error locking field. They are set by the occurrence of the conditions mentioned. The setting of either bit will cause the UBA to make an interrupt request on the SBI if the SBI to Unibus error interrupt enable bit <03> (SUREIE) in the control register is set. The setting of either UBSTO or UBSSYNTO will load the Failed Unibus Address Register, thus storing the high 16 bits of the UNIBUS address identified with the failure. The Failed Unibus Address Register will remain locked until the UBSTO and UBSSYNTO bits are cleared.

NOTE: 1. Seven of the above bits (RDTO, RDS, CXTER, CXTMO, DPPE, IVMR and MRPF) form an error locking field. If any of these bits is set, the field is locked, thereby preventing the setting of other bits within this field, until the bit indicating the error is cleared. The failed map entry register (UMER) is also locked and unlocked with this field. The setting of these bits will cause the UBA to initiate an interrupt request if the interrupt enable bit for the Unibus to SBI data transfer error field (SUREIE) in the control register is set.



This register is made up from the KMUX register (see EBD2, EB5) and EBOX) and other EBox status bits (see EB3B and EB4).

<31:27> CONSOLE ECC CORRECTION REQUESTS
The bits in this field are set by the ESM in order to interrupt the Console for Control Store or Dispatch RAM correction.

When set, these bits will cause an immediate Console interrupt. The ESM will loop waiting for the Console to set "DONE" in RBUPC. When correction is complete, control is returned to the EBox by setting "DONE". The ESM will then clear the correction request EBOX <31:27> and thus release the Console interrupt.

<31> FBM CONTROL STORE CORRECTION REQUEST
EB5E FBOX FBM CS PE
The Console will attempt to correct the parity error and then return control to the ESM by setting the "DONE" bit <07> in RBUPC.

<30> FBA CS CORRECTION REQUEST
EB5E FBOX FBA CS PE
The Console will attempt to correct the parity error and then return control to the ESM by setting the "DONE" bit <07> in RBUPC.

<29> FBOX DRAM CORRECTION REQUEST
EB5E FBOX DRAM PE
The Console will reload the FBox Dispatch RAM and then return control to the ESM by setting the "DONE" bit <07> in RBUPC.

<28> IBOX DRAM CORRECTION REQUEST
EB5E IBOX DRAM PE
The Console will attempt to correct the parity error and then return control to the ESM by setting the "DONE" bit <07> in RBUPC.

<27> IBOX CS CORRECTION REQUEST
EB5E IBOX CS PE
The Console will attempt to correct the parity error and then return control to the ESM by setting the "DONE" bit <07> in RBUPC.

<26:24> RESERVED

<23:21> STACK FRAME REVISION
The stack frame revision is written by the error handling microcode to indicate how many times the stack frame has been revised. The previous stack frame was revision 0, and the current stack frame is revision 1.

<20> PERFORM MONITOR ENABLE
EBE5 PERFORMANCE MONITOR ENABLE
This is an architecturally defined bit which is controlled by system software. It is wired to a CPU backplane pin (slot C9 Pin 40?) and allows system performance to be monitored externally. See the PMK description for further information.

<19:16> PROCESS ABORT REASON CODE
If EMMSYR <15>, PROCESS ABORT, is set, EBCS <19:16> is the reason why the faulted instruction cannot be retried.

0001 Unrecoverable GPR parity error
0010 EBox WBus parity error
0011 All IBox PC's are invalid
0100 EBox failed to detect CPBus byte parity error
0101 GPR parity error that did not result in an MBox fatal error
0110 Rlog parity error

<15> MBOX FATAL ERROR
ERR& MBOX FATAL ERROR
Set via EBox Port Status Line 0. Indicates that the MBox detected one of the following Fatal Error conditions:

ERR7 WR DAT PE & WRITE ERR	ERR2 CP ID PE
ERR2 IAC PE & MSIT	ERR6 CP SLEP ERR
ERR5 DMA PE	ERR8 CP MEM PRE
MCC3 CVM PIL ERR	

<14> MBOX INTERUPT PENDING
ERC3 MBOX IRIR LVL3
The MBox generates an interrupt request when it detects an error of any kind (excluding TB parity errors). This bit is set by the EBox on the third occurrence of T3 after it receives MBox Interrupt and is usually handled at IRD time.

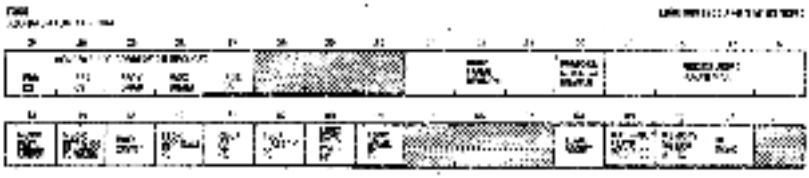
<13> IBOX ERROR
EBEA IBOX ERR L2H
Set by the IBox when the IBox reports one of the following error conditions:

ICBC IDSAM PE	IDP6 IPRUX PE	ICA7 ICS PE
ICB7 ILV MOUR	IDP6 IAPUX PE	ICBC IBUF PE
ICB6 BLOC PE		

<12> EBOX MEMORY CONTROL FIELD RAM PARITY ERROR
XHR2 PMER PE FLAG
Set when the XBox detected a parity error in the Memory Control Field (MCF) RAM.

<11> EBOX CONTROL STORE PARITY ERROR
EBD2 SCS PE FLAG
Set when the EBox detected a Control Store Parity Error. Setting this bit results in a immediate trap to the Console for ECC Correction. When the Console finishes correcting the error it will force the EBox to trap to ESK (Vector 8).

<10> EBOX MICRO-STACK PARITY ERROR
EBD2 USTK PE FLAG
Set when the EBox detected a parity error when it popped the last entry off the micro-stack.



<09> EBOX DATA PATH PARITY ERROR
EBD2 EDP PE FLAG
Set when the EBox detected either an Operand parity error or a Result parity error.

<08> EBOX MBUS PARITY ERROR
EBD2 MBUS PE FLAG
Set when the EBox detected a MBus Parity Error while it was driving the bus.

NOTE

1. Writing 1 to this bit clears bits <15>, <12:09> and <8>.

2. This bit has a different use in diagnostic mode.

<04:01> VMS ABORT FLAGS
VMS uses these flags, in part, to determine whether or not the error is recoverable.

<04> EBOX ABORT
EBD3 EB ABORT FLAG
Set when the EBox detected an MCP RAM Parity Error or a Result Parity Error. Both cases are non-recoverable.

<03> MACHINE STATE MODIFIED
EBD3 SVA MOD FLAG
Set by Microcode via the UMISC field. Indicates that the state of the machine has been modified such that, should an error occur while this bit is set the operation currently executing in the EBox cannot be retried.

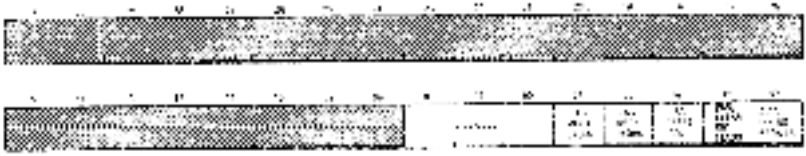
<02> MEMORY OR GPR WRITE
EBD3 MEM WR? FLAG
Set when a Memory or GPR write operation was in progress when the error occurred.

<01> IO READ
EBD3 IO RD FLAG
Set when the error involved an I/O register read. Since some I/O registers automatically clear after being read the contents of the I/O register may have been lost.

<00> RESERVED

<(1:100)> LAST EBOX WORD WRITTEN
This register contains a copy of the last word that the EBox wrote to the MBox. This word is obtained by popping the last word off the Scratch Pad Stack (MScratch Location F0).

<(1:100)> SECOND TO LAST EBOX WORD WRITTEN
This register contains a copy of the second to last word that the EBox wrote to the MBox. This word is obtained by popping the second to last word off the Scratch Pad Stack (MScratch Location F1).



The EBox Diagnostic/Maintenance Control Register (EDMC) is a write only register which shares the same MBus register address as the EBox Data Path Error Register (EDPEL), a read only register. EDMC implements up to 8 bits which can be used by EBox microdiagnostics to control hardware in the CPU.

A Master Reset or Diagnostic Reset causes EDMC to be written with all zeros, thus clearing (X) DIAG, CLR FLAG, RST IBOX, and MB LADR <07:06>.

<07:05> RESERVED

<04> DIAGNOSTIC MBOX MICRO ADDRESS 07
EBD DIAG MBOX UADR 7 L <Pin A06-69> V$A102
Diagnostic MBox Microaddress bit 7. This bit is used by diagnostics to control bit 7 of the MBox Microsequencer Address.

<03> DIAGNOSTIC MBOX MICRO ADDRESS 06
EBD DIAG MBOX UADR 6 L <Pin A06-71> V$A101
Diagnostic MBox Microaddress bit 6. This bit is used by diagnostics to control bit 6 of the MBox Microsequencer Address.

<02> DIAGNOSTIC RESET IBOX
EBD DIAG RST IBOX H <Pin A06-87> V$D110
Diagnostic Reset IBox. This bit initiates an IBox reset.

<01> DIAGNOSTIC CLEAR CYCLE IN PROGRESS FLAG
EBD4 DIAG CLR FLAG H V$C198
Diagnostic Clear Cycle in Progress Flag. The assertion of this bit clears the Cycle in Progress Flags in the EBox Control Logic when they are clocked at the end of a machine cycle.

<00> DIAGNOSTIC INHIBIT MSTALLS
EBD4 DIAG INH STALLS H V$C177
Diagnostic Inhibit EBox Stalls. The assertion of this bit prevents the EBox Microsequencer and EBox Data Path from stalling when EBox Stall is asserted. The bit does not inhibit the assertion of copies of EBox Stall used by the MBox, IBox, MBox, or EBox Control Logic or the copy which serves as a branch condition to the EBox Microsequencer.



This register is made up from the nibble registers in the EDP MCA.

<31:28> RMUX BYTE ID ERROR
EDP4 RMX ERR BYTE <3:0> (RDPH)
This field is only valid when either bit <07> or bit <00> is set. This field indicates the byte(s) that were associated with the RMux Parity Error.

<27:24> AMUX Byte in ERROR
EDP3 AMX ERR BYTE <3:0> (RDPH)
This field is only valid when either bit <01> is set, or when bit <02> is set, or when bit <03> is set and bits <00:02> and <04:07> are reset. This field indicates the byte(s) that were associated with the AMUX Parity Error.

<23:16> EBOX GPR PARITY ERROR ADDRESS
If BITMAPS <25> [EBox GPR B PE] or <23> [EBox GPR A PE] is set, these bits indicate the failing GPR address.

<15:12> VMD BYTE IN ERROR
EDP5 VMD BER BYTE <3:0> (RDPH)
This field is only valid when bit <05> is set and bit <08> is reset. It indicates the byte(s) that were associated with the VMbuff parity error.

<11> WREGISTER PARITY ERROR
EDP6 WREG BER (RDPH)
Indicates that the parity at the input to the WReg Mux did not match the parity generated at the output of the WReg. The inputs to the WReg Mux are:

- The AMux and BMux for WReg Shift Operations.
- The BMux for WReg Format Operations.
- The ALU for post ALU Wreg Shift Operations.

NOTE

This error will also cause a RESULT PE.

<10:09> RESERVED

<08> EDP MISCELLANEOUS PARITY ERROR
EDP8 EDP MISC (RDPH)
Indicates that the ALU was the input to the VMQMUX when a parity error was detected at the VMQMUX output. When this bit is set the contents of bits <12:15> are not valid. This error will also cause a RESULT PE.



<07> EBOX WBUS PARITY ERROR
PDP8 B WBUS (EDPH)
Indicates the WBus was the input to the EBox when a parity error was detected at the EBox output. Bits <28:31> indicate the byte(s) in error.

<06> EBOX OPBUS PARITY ERROR
PDP8 B OPBUS (EDPH)
Indicates that the OPBus (which is protected by longword parity) was the input to the EBox when a EBox parity error was detected at the EBox output. When this bit is set the contents of bits <28:31> are not valid.

<05> RESULT PARITY ERROR
PDP% RSLT CHK (EDPH)
If neither WRMG PE <11> nor FPP MISC <18> are set, this bit indicates that the VAQSAV Register was the input to the VAQMUX when a VAQBOX parity error was detected. Otherwise, this bit is the "OR" of all three of those conditions.

<04> RESERVED

<03> OPERAND PARITY ERROR
PDP8 OPER CHK (EDPH)
If bits <02:00> and <07:06> are reset, then this bit indicates that the VAQSAV Register was the input to the AMux when an AMux parity error was detected. Otherwise, this bit indicates that one of the following errors were detected:

BMux GPR B PE	AMux GPR A PE
BMux WBus PE	AMux WBus PE
BMux OPBus PE	

<02> EBOX GENERAL PURPOSE REGISTER A PARITY ERROR
PDP8 A RAM (EDPH)
Indicates that Scratch Pad-A was the input to the AMux when a parity error was detected at the AMux output.

<01> AMUX WBUS PARITY ERROR
PDP8 A WBUS (EDPH)
Indicates that the WBus was the input to the AMux when a parity error was detected at the AMux output.

<00> EBOX GENERAL PURPOSE REGISTER B PARITY ERROR
PDP8 B RAM (EDPH)
Indicates that Scratch Pad-B was the input to the BMux when a parity error was detected at the output of the BMux.



This status word contains a modified copy of the Error Handling Status Register (EHSR) which is stored in Scratch Pad Location 0F8. The Error Handling Microcode uses EHSR to keep track of status during the error handling process.

In addition, two other Microcode Routines use the EHSR to pass status to the EBOX Routine. The Interrupt Handling Microcode Routine uses bit 31 to report MBox error interrupts and the FBox Problem Handling Routine uses bit 28 to report FBox hardware errors.

The EBOX copies the contents of EHSR into EBox Scratch Pad location 18 just before it sets VMS ENTERED and clears EBOX ENTERED. The EBOX then calls the Interrupt Exception Microcode to push the stack frame onto the interrupt stack and call the Machine Check Handler.

<31> MBOX SERVICE REQUEST
If the Interrupt Handling Microcode determines that it was called to handle an MBox Error Interrupt it will set this flag and call the EBOX. The EBOX will test this flag and, if set, the EBOX will process the MBox Error.

<30> EBOX ENTERED
This flag is used by the EBOX to detect a double error trap condition. That is, some cases when a second error trap occurs before the EBOX Routine is able to finish processing the first error and pass control to VMS.

This flag is checked each time the EBOX routine is entered. If the flag is clear (which is the expected case) then the EBOX Routine will set this flag and process the error in the normal manner. If the flag is set, however, indicating that the EBOX Routine was in the process of handling an error when it was called to handle a second error, then one of two things will happen:

1. If the second error was detected by either the IBox or the MBox Fatal Error Detection circuitry, then the EBOX Routine will loop at UPC 41. This in turn will cause a Loop Alive Fail condition and the Console will Print the following message and Snap Shot the state of the system.

   "Attempting to save machine state due to" "MACHINE DOUBLE ERROR"



2. If the second error was detected by the IBox then the ESM will put a code of 5 in CSM.STATUS (Scratch Pad location 70) and call the CSM.ENTRY.DE Routine. This will result in a Keep Alive Fail Condition and the console will print the following message and Snap Shot the state of the system.

   "Attempting to save machine state due to" "CPU ERROR HALT"

   NOTE

   Because the state of EHSR is saved in SSerator 14 before the ESM Routine clears EHR ENTERED, this flag will always be set in the copy of the Stack Frame saved by the VMS MCHK Handler.

<29> VMS ENTERED
This flag is similar to the ESM ENTERED flag. It is used to detect the case where the VMS Machine Check Handler is in the process of handling an error when a second error is detected.

The ESM Routine sets this flag just before it calls the Version Check Handler. The Machine Check Handler processes the errors and clears this flag just before it executes an REI to continue the operation (or a BUGCHECK to halt the operation).

If a second error trap occurs while the Machine Check Handler is still processing the first error, then the ESM Routine will process the error in the normal manner. That is, build a Stack Frame, clear the error condition, roll back the PCs and determine if it should call VMS.

However, since the VMS ENTERED flag is set indicating that VMS was processing an error when a second error was detected, the ESM will not call VMS. Instead, it will put a code of 5 in CSM.STATUS (IBox Scratch Pad Location 70) and call the CSM.ENTRY.DE Routine. This in turn will result in a Keep Alive Fail Condition and the console will print the following message and Snap Shot the state of the system.

"Attempting to save machine state due to" "CPU ERROR HALT"

<28> FBOX SERVICE REQUEST
The microroutine that handles FBox Problems will set this flag if it determines that the problem was caused by an FBox hardware error. The routine will then call the EBox Routine to process the error. The CHM will test this flag, to determine if it was called to handle an FBox error.

<27> FIX FBOX GENERAL PURPOSE REGISTER PARITY ERROR
Set by the CHM when it starts to correct an FBox GPR parity error.

<26> FIX EBOX GENERAL PURPOSE REGISTER B PARITY ERROR
Set by the CHM when it starts to correct an EBox GPR B parity error.

<25> FIX EBOX GENERAL PURPOSE REGISTER A PARITY ERROR
Set by the CHM when it starts to correct an EBox GPR A parity error.

<24> FIX IBOX GENERAL PURPOSE REGISTER PARITY ERROR
Set by the CHM when it starts to correct an IBox GPR parity error.

<23> FIX FBM CONTROL STORE PARITY ERROR
Set by the CHM when it starts to correct an FBM Control Store parity error.

<22> FIX EBA CONTROL STORE PARITY ERROR
Set by the CHM when it starts to correct an EBA Control Store parity error.

<21> FIX FBOX DRAM PARITY ERROR
Set by the CHM when it starts to correct an FBox Dispatch RAM parity error.

<20> FIX IBOX DRAM PARITY ERROR
Set by the CHM when it starts to correct an IBox Dispatch RAM parity error.

<19> FIX EBOX CONTROL STORE PARITY ERROR
Set by the CHM when it starts to correct an IBox Control Store parity error.

<18> MBAR SAVED
Indicates that the MBox Error Address Register (MBAR) was saved (in Scratch Location: 0B) as a result of the last MBox Error Register Full trap.

<17> PROCESS ABORT
The EBox microcode detected a condition which prevents a retry of the failing instruction. See EBOX <15:15> for the reason code.

<16> RESERVED



<15:08> MICRO TRAP VECTOR ADDRESS
Contains the vector address through which the EBOX was entered.

VECTOR	REASON
2	FBox Error (Called by FBox Interrupt Handler)
4	EBox detected a Process Abort condition during a MBox Error Register Full Microtrap.
6	MBox Error (Called by MBox Interrupt Handler)
8	BBox Error
8	MBox Fatal Error.
8	TB PE Error (EBox Port Request only)
10	IBox Op-Port-Write and IBox Error
10	IBox Op-Port-Write and TB Parity Error
10	EBox IRD Read and IBox Error
10	EBox IRD Read and TB Parity Error
10	EBox Fork and IBox Error
10	EBox Fork and TB Parity Error
10	EBox ID Read and IBox Error
10	EBox String Read and IBox Error
10	EBox String Read and TB Parity Error
13	IBox Sync Failure
1F	Diag Unwind Failure

<07:00> REASON CODE
This field is supplied by the VMS Machine Check Handler. Therefore, this field will be valid only after the Machine Check has been processed by VMS. This field will not be valid for Stack Frames extracted directly from the EBox Scratch Pad RAMs or the Interrupt Stack. See <07:00> for the actual Reason Code.

<07> EBOX INTERRUPT ENTRY VALID
Indicates that the Stack Frame was generated as a result of an EBOX IE interrupt.

<06> SBI SUMMARY ENTRY VALID
Indicates that a copy of the SBIA Error Summary Register has been appended to the end of the Stack Frame.

<05> SBIA FULL REPORT FOLLOWS
Indicates that a full SBIA Error Log entry follows this entry in the System Event File (SYSEVENT.SYS).

<04> RESOURCES TURNED OFF
Indicates that VMS disabled either the PBus or one of the Caches.

<03:00> PRIMARY ERROR CODE
This field indicates which Box detected the error.

Code	Box Detecting Error
001	PBox
010	EBox
011	IBox
100	MBox (Fatal Error)



<31:27> EBOX IS FIXING
The KRM sets the appropriate bit in this field just before it sets the corresponding bit in EPCS <31:27>. Setting a bit in EBCS <31:27> causes the EBox to interrupt the Console for Control Store or Dispatch RAM correction.

<26> MDAR SAVED
When the KRM is called to handle an MBox Error Register Full (EERF) micro trap, it saves MEAR in EEVR DS and sets this bit. Later, when the EHM is servicing the MBox interrupt associated with the EERF micro trap, it checks this bit to determine whether it should get MEAR from the MBox or EEVR DS.

<25> PROCESS ABORT
The KHM sets this bit if it determines that: 1) An MBox CPE Parity Error failed to result in an MBox Fatal Error, or 2) the EBox failed to detect bad data on the CPBus. If this bit is set, VMS will either Bugcheck the user or the system.

<24> MBOX TRAP TO 4
This bit indicates the occurrence of an MBox trap to 4. When running with the the IPL above 1D this may be the only sign of an SBE. It is used by VMS to check for single bit errors.

<23:16> MICRO-TRAP VECTOR ADDRESS
The EHM saves the entry level trap vector address in this field. The vector addresses are:

VECTOR	REASON
2	MBox error (Called by MBox interrupt handler)
4	EHM detected a process abort condition during a MBox SBE micro-trap
6	MBox error (called by MBox interrupt handler)
8	EBox error
A	MBox fatal error
C	IS parity error (EBox port request only)
10	EBox Op-Port-Write and IBox error
12	IBox Op-Port-Write and TP parity error
14	MBox IRD read and TPar error
16	EBox IMD read and TB parity error
VECTOR	REASON (cont)
18	EBox Fork and IBox error
1A	MBox Lock and TB parity error
1C	EBox IB read and IBox error
1E	EBox string read and MBox error
20	EBox string read and TP parity error
22	MBox sync failure
24	RLog unwind parity error

<15:08> RESERVED

<07> MBOX SERVICE REQUEST
If the interrupt handling microcode determines that it was called to handle an MBox error interrupt, it will set this flag and call the EHM. The EHM will test this flag and, if set will process the MBox error.

<06> EHM ENTERED
This flag is used by the EHM to detect a double error trap condition. That is, those cases when a second error trap occurs before the EHM routine is able to finish processing the first error and pass control to VMS.

This flag is checked each time EHM is entered. If the flag is clear (which is the expected case), then EHM will set this flag and process the error in the normal manner. If the flag is set, however, indicating that EHM was in the process of handling an error when it was called to handle a second error, then one of two things will happen:

1. If the second error was detected by either the EBox or the MBox (intel error detection circuitry, then EHM will loop at UPC 21. This in turn will cause a Keep Alive Fail condition and the Console will print the following message and capture (Snap Shot) the state of the system:

" Attempting to save machine state due to"
"MACHINE DOUBLE ERROR"

2. If the second error was detected by the FBox then EHM will put a code of 5 in CSM.BIAfu (BSC: C0) and call the CSM.ENTRY.PT routine. This will result in a Keep Alive Fail Condition and the Console will print the following message and Snap Shot the state of the system:

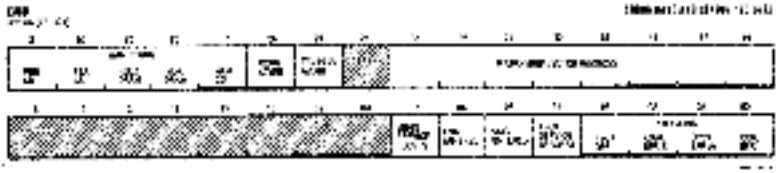
" Attempting to save machine state due to"
"FBOX ERROR 5417"

NOTE

When EHM finishes processing the error, it will set the VMS ENTERED flag, clear this flag and call the VMS Machine Check Handler.

<05> VMS ENTERED
This flag is similar to the EHM ENTERED flag. It is used to detect the case where the VMS Machine Check Handler is in the process of handling an error when a second error is detected.

EHM sets this flag just before it calls the Machine Check Handler. The Machine Check Handler processes the errors and clears this flag just before it executes an REI to continue the operation (or a BUGCHECK to halt the operation).



If a second error trap occurs while the Machine Check Handler is still processing the first error, then the EHM routine will process the error in the normal manner. That is, build a Stack Frame, clear the error condition, roll back the PC's and dekexaine if it should call VMS.

However, since the VMS ENTERED flag is set (indicating that VMS was processing an error when a second error was detected), EHM will not call VMS. Instead, it will put a code of 5 in CSM.STATUS (ESC: C0) and call the CSM.ENTRY.OK routine. This in turn will result in a Keep Alive Fail condition and the Console will print the following message and capture (Snap Shot) the state of the system:

" Attempting to save machine state due to"
"CPU ERROR HALT"

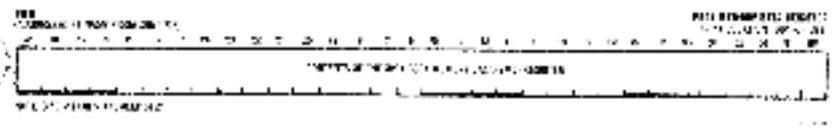
<0:6> FBOX SERVICE REQUEST
The micro-routine that handles Fbox problems will set this flag if it determines that the problem was caused by an Fbox hadrware error. The routine will then call the EHM routine to process the error. The EHM will test this flag to determine if it was called to handle an Fbox error.

<03> FIX FBOX GPR PE
Set by the EHM when it attempts to correct an FBox GPR parity error.

<02> FIX EBOX GPRB PE
Set by the EHM when it attempts to correct an EBox GPR B parity error.

<01> FIX EBOX GPRA PE
Set by the EHM when it attempts to correct an EBox GPR A parity error.

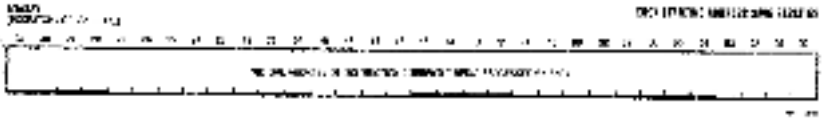
<00> FIX IBOX GPR PE
Set by the EHM when it attempts to correct an IBox GPR parity error.



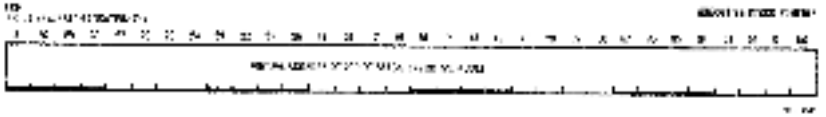
<31:00> TBUK BACK MEMORY DATA REGISTER
The contents are displayed in bit <31:00>.

Additional Notes:
(1) CSK Utility "Examine_TBUK Miscellaneous_Register", is used to read the contents of the EMD Register. Basically, the OPSEL field selects the EMD register to drive the OPBUS which in turn is loaded into an ESP location.

(2) The EMD register is not visible on any backplane pin nor on the SBI visibility path.



<31:00> CURRENT PC FOR EXECUTION UNIT (PBOX)
This register contains the address (Macro PC) of the instruction that the EBox or FPA is currently processing.



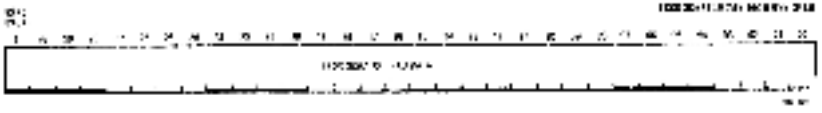
<31:00> EXECUTIVE STACK POINTER
Contains the stack pointer to be used when the current access mode field in the PSL is 1 (Executive Mode).

BSPA/BSPD
These registers allow VMS to access the EBox GPR/SP RAMS using macro instructions (MFPR/MTPR). For example, during machine checks for array single bit errors, the VMS machine check handler would access the EBox state saved in RAM locations 25 thru 28 (hex). Executing macro instructions between the MTPR and MFPR will yield unpredictable results. The BSPA must be loaded with the RAM address followed immediately by reading the data from BSPD.

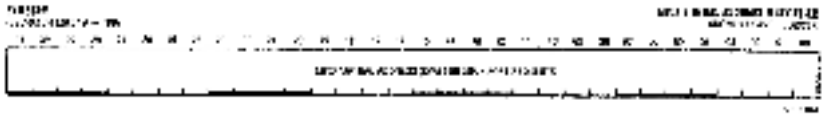
The reason for the above scenario is that for array single bit errors, the Stack Frame is not pushed on the interrupt stack. Therefore, the VMS machine check handler must access the GPR/SP explicitly.



This write only register is loaded with the address of an EBox scratchpad location to be accessed.



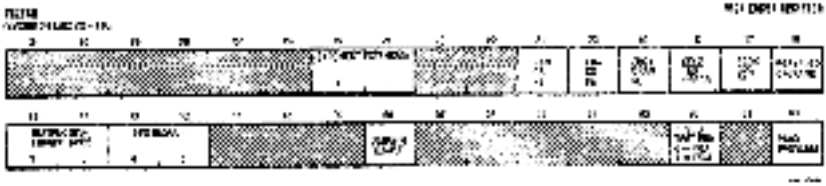
This read only register contains the data requested by the MFPR to BSPD. If BSPA is not loaded prior to accessing the data, unpredictable results will occur.



EVMQSAV (EBox Virtual Memory Address/Multiplier Quotient Save Register).

During EBox port tracking this register contains the virtual address that was acknowledged by the MBox (PA ACK). During normal operation, this register is used to temporarily store partial EBox results.

<31:00> May contain an <MBox Virtual Address> or a partial calculated EBox result depending on the operation of the EBox.



The BDM builds this register in the FBox Scratch Pads by reading three FBox registers: 03, 02, and 01.

<31:26> RESERVED

<25:24> EXPONENT EXTENSION <01:00>
FBRD EXT <1:0> (FA01)
This field, which is an extension of the exponent field, indicates overflow and underflow conditions as follows:

FIELD	CONDITION
00	Normal
01	Overflow
10	Underflow
11	Undefined

<23:22> RESERVED

<21> FBM CONTROL STORE PARITY ERROR
MPR5 CS PAR ERR (FR07)
Set when the FBM module detected an FBM Control Store Parity Error. When set, the CS Address is latched in the FBM MSD MCA.

<20> FBA CONTROL STORE PARITY ERROR
ACC4 RAM PERR (FA13)
Set when the FBA module detected an FBA Control Store Parity Error. When set, the Control Store Address is latched in the FBA MSD MCA.

<19> FBOX DRAM PARITY ERROR
MCB3 FDRAM PAR ERR (FR11)
Set when the FBM module detected a Dispatch RAM Parity Error. Neither the DRAM address nor the DRAM data are latched.

<18> SELF TEST ERROR
FBR3 SELF TEST ERROR (FA01)
Set when the FBox detected an error while running the Self Test. Bit <02> will indicate whether the error was associated with the FBA or FBM Module.

<17> FBOX GENERAL PURPOSE REGISTER PARITY ERROR
FBR4 GPR ERROR (FA01)
Set when the FBA Module detected a GPR Parity Error. The specific byte in error cannot be identified.

<16> RESERVED OPERAND
FBR6 RESERVED OP (FA01)
This bit generally indicates a software problem and has no significance in the context of a hardware error. It indicates that one of the operands received by the FBox had a negative sign and an exponent of zero.

<15:14> INSTRUCTION FORMAT CODE <01:00>
FBR5 FORM <01:00> (FA01)
This field indicates the format of the instruction that the FBox was executing when the FBox status was saved in this register.

CODE	FORMAT
00	F
01	G
10	D
11	L or Q

<13:12> FPA RESULT <01:00>
FXP6 FRES <5:6> (FA01)
This field has no significance in the context of an error. It represents exponent result bits <1:12>.

<08> DENORM RESULT
FBR1 DENORM (FA01)
This bit has no significance in the context of errors. It indicates that a rounding operation resulted in a "carry out".

<07:03> RESERVED

<02> IF SELF TEST ERROR (0=FA/FM=1)
FBR4 WHOS DOO (FA01)
This bit has a double meaning. During normal operation this bit indicates that the divisor was equal to zero. If SELF TEST ERROR is set, however, this bit indicates which FBox module detected the Self Test error (0=FA/FM=1).

<01> RESERVED

<00> FBOX PROBLEM
FBR1 FBOX PROBLEM (FA01)
Set when the FBox detected one of the following conditions:

Exponent Extension Problem	bits <25:24>
GFE Parity Error	Bit <17>
FBM Control Store Parity Error ...	Bit <21>
FBA Control Store Parity Error ...	Bit <20>
FBRAM Parity Error	Bit <19>
Self Test Error	Bit <18>
Divide by Zero	Bit <02>
Denormalize Result	Bit <00>
Reserved Operand	Bit <16>



This register is a combination of the IBox Error register (IBER) and the SBox Diagnostic Maintenance register (SDMS).

<31> RESERVED

<30:29> IBOX AMUX BYTE IN ERROR CODE
EBPA IAMUX PE <1:0> LTH
Indicates the most significant byte associated with the IBox AMux parity error.

Code	Byte
00	0
01	1
10	2
11	3

<28> IAMUX SOURCE (0=GFE/1=WBUS)
EBEA IMUX DATA LTH
Indicates that the WBus was the input to the IBox AMux when an error was detected at the output of the IBox AMux.

<27> RESERVED MODE DETECTED
EBEA RSV MODE LTH
Indicates that an operand specifier attempted to use an addressing mode that is not allowed in the situation in which it occurred.

<26> IBOX SMUX PARITY ERROR
EBEA ISMUX PE LTH
Indicates that a parity error was detected at the output of the IBox SMux. The input to the SMux was either the IBuffer or the IMD latch.

<25> INST BUFFER PARITY ERROR
EBPA IBUF PE LTH
Indicates that a parity error was detected on either the OPCode Bytes (Byte 0 or Byte 1 in the IBuffer), or on the byte selected by the BMode Finder (IBGEN) during optimization.

<24> RLOG PARITY ERROR
EBEA RLOG PE LTH
Indicates that a parity error was detected while unwinding the RLog.

<23> IBOX AMUX PARITY ERROR
EBEA IAMUX PE LTH
Indicates that a parity error was detected at the output of the IBox AMux. The input to the AMux was either the WBus or a GPR. See IBER <30:29> and <28>.

<22> IBOX DRAM PARITY ERROR
EBEA IDRAM PE LTH
Indicates that a parity error was detected on the DRAM data.

<21> IBOX CS PARITY ERROR
R/W/A TCS PK CIM
Indicates that a parity error was detected on the IBox Control Store data.

<20:16> RESERVED

<15:08> NORMAL REGISTER BITS <D15:D08>
See BPSA

<15> RESERVED

<14> ENABLE EBOX MICRO TRAP LOGIC
EBDJ EN XTRAP
Enables the EBox microtrap mechanism. This in turn allows CPU error reporting.

<13:11> MICRO TRAP PRIORITY LEVEL
EBDX XTRP <2:0>
This field indicates the priority of the last microtrap request.

Priority	Microtrap Type
0	EBox Read/Write Microtrap
1	SP Write Microtrap
2	IBox Error Microtrap
3	Misc Microtrap
4	Fork Microtrap
5	IMD Read Microtrap
6	ID Read Microtrap
7	SIRISC Read Microtrap

<10> OPBUS SOURCE (0=ID/1=IMD)
EBDS SRC IMD CVE.3
This bit is only valid when rmbox <09:08> = 3. When set, this bit indicates that the OPBus came from IMD register. When cleared, it indicates that the OPBus data came from ID register.

<09:08> OOPSEL <1:0>
EBDS DOPSEL <1:0>
This field indicates the OPBus data source.

<1:0>	Data Source
0	IBox Register Select
1	Operand Source is EBD
2	Operand Source is IBUFFER
3	Operand Source is IMD or ID Registers (See Bit 10 above)

<07:03> RESERVED

<02> PSA VALID
This bit is set by the MBM if the IBox ESA VALID bit is set.

<01> ISA VALID
This bit is set by the MBM if the IBox ISA VALID bit is set.

<00> CPU VALID
This bit is set by the EBM if the IBox CPU valid bit is set.



<U1:UD> IBOX GPR ADDRESS
Lines are generated by the IBOX (Print:ICB0) and are distributed to the EBOX (Print:EDPC) and the FBOX (Print:PA12) when OPTIMISATION is done. Note that the IBGPR lines are also latched and distributed internally within the IBOX to perform a variety of functions.

NOTE

1. The IBGPR register (read only) is accessed via the following console command:

   >>>EXAM IBGPR<cr>

2. The IBGPR address lines are also available on the SDM visibility bus (all signals are generated on ICB0 and are terminated on the FBA module).

IBGPR BIT	SIGNAL NAME	SDM SYMBO	B/P PIN(ICB0)
3	ICB IBGPR 3 H	V$0107	AC12A60
2	ICB IBGPR 2 H	V$0108	AC12A54
1	ICB IBGPR 1 H	V$0118	AC12A78
0	ICB IBGPR 0 H	V$0119	AC12A60



<31> ERROR
Whenever ICR overflows, if INT is already set, then ERR is set. Thus, VAX indicates a missed clock tick. An attempt to set this bit via MTPR clears INT.

<30:08> MBZ
Must be zero.

<07> INTERRUPT
Set by hardware every time ICR overflows. If IE is set then an interrupt is also generated. An attempt to set this bit via MTPR clears INT, thereby reenabling the clock tick interrupt (if IE is set).

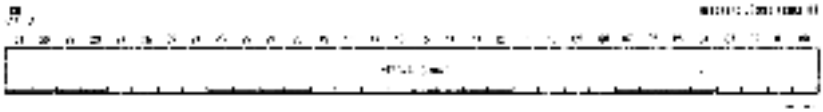
<06> INTERRUPT ENABLE
When set, an interrupt request at IPL 16(16) is generated every time ICR overflows (INT is set). When clear, no interrupt is requested. Similarly, if INT is already set and the software sets it, an interrupt is generated (i.e., an interrupt is generated whenever the function (IE and INT) changes from 0 to 1). At boot time this bit is cleared.

<05> SINGAL
A write only bit. If RUN is clear, each time this bit is set, ICR is incremented by one.

<04> TRANSFER
A write only bit. Each time a 1 is written to this bit, XICR is transferred to ICR.

<03:01> MBZ
Must be zero.

<00> RUN
When set, ICR increments each microsecond. When clear, ICR does not increment automatically. At bootstrap time, RUN is cleared.

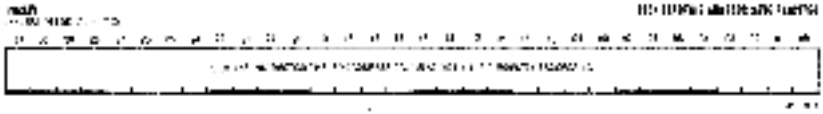


<31:00> INTERVAL COUNT
The Interval register is a read only register incremented once every microsecond. It is automatically loaded from NICR upon a carry out from bit 31 (overflow) which also interrupts at IPL 16(16) if the interrupt is enabled.

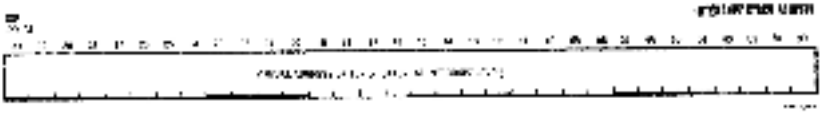


<31:05> MBZ
Must be zero.

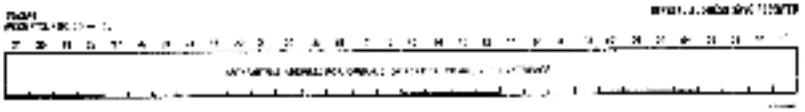
<04:00> INTERRUPT PRIORITY LEVEL
Writing to the IPL with the MTPR instruction will load the processor priority field in the Program Status Longword (PSL), that is, PSL <20:16> is loaded from IPL <04:00>. Reading from IPL with the MFPR instruction will read the processor priority field from the PSL. On writing IPL, bits <31:05> are ignored, on reading IPL bits <31:05> are returned as zero.



<31:00> CURRENT PC FOR ADDRESS CALCULATION UNIT
This register contains the address (Macro PC) of the instruction that the IBox Address Calculation Unit is currently processing.

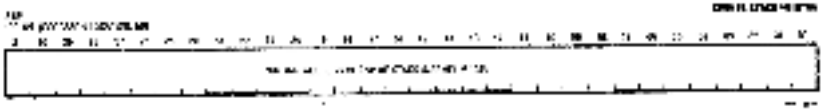


<31:00> VIRTUAL ADDRESS OF TOP OF STACK
This is the stack pointer for the interrupt stack. Unlike the stack pointer for process context stacks, which are stored in the hardware PCB, the interrupt stack pointer is stored in an internal register. Refer to PSL to determine which SP is currently being used. The ISP is in use if we are in kernel mode and if PSL <26>, Interrupt Stack, is set.



IVASAV (IBox Virtual Address Save) - This register contains the last virtual address that was calculated by the Address Calculation Unit and acknowledged by the MBox with DA ACK. Therefore, IVASAV will contain the Virtual Address associated with either the current operand fetch cycle, or the current result store cycle.

<31:00> Last Virtual Address used by the IBox for either an Operand Fetch or Result Store.

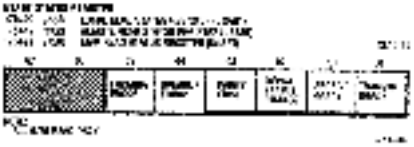


<31:00> KERNEL STACK POINTER
Contains the stack pointer to be used when the current access mode field in the PSL is 0 and IS = 0 (Kernel Mode).

VAX 8600/8650 REGISTER DESCRIPTION
USART RECEIVE HOLDING REGISTER
USART STATUS REGISTERS



<07:00> RECEIVED CHARACTER
When a complete character has been received on the USART serial line, it is moved from the USART shift register to the USART Receive Holding Register.



<07:06> RESERVED

<05> FRAMING ERROR
Indicates that the character just received was not framed by the correct number of stop bits.

<04> OVERRUN ERROR
Indicates that a data character was not read by the T-11 program before the next one was assembled and loaded in the Receive Holding Register.

<03> PARITY ERROR
Indicates that bad parity was received.

<02> DEVICE STATUS CHANGE
The transmitter has completed serialization of character in Transmit Holding Register, or DSR or RXD have changed state.

<01> RECEIVER READY
RXRDY indicates to the T-11 program that it may read the USART holding register to take the data.

<00> TRANSMIT READY
TXRDY is set by the USART, signifying that the USART is ready to accept the character. This generates a T-11 interrupt request.



The 2-11 program loads the character in the USART Transmit Holding Register. TXRDY is negated. When the character previously loaded (if any) has been transmitted, the character is moved to an internal shift register and transmitted on the serial line. When the character is moved to the shift register, TXRDY is again asserted, indicating that the transmit holding register is empty and ready to accept another character.



NOTE: One address is used to write a command register and another address to read the status register.

<07:06> SUBMODE

00 = Normal
01 = Auto Echo
10 = Local Loopback
11 = Remote Loopback

<05> REQUEST TO SEND (RTS)
For the remote line only, Request To Send (RTS) is asserted in the USART Command register. This signal is set by the SVC program when it is ready to place or answer a call on the remote line.

<04> ERROR RESET (ER)
The setting of this bit clears the Parity Error, Overrun Error and Framing Error bits (Read Status Register bits <05:03>). Error Reset must be performed when Receive Enable is set.

<03> SEND BREAK CHARACTER (SBRK)
This is a programmable bit and when set, the output of the USART transmitter, TX DATA, goes to a logical zero state. This condition will generate a continuous spacing signal to be transmitted. When this flag is cleared, normal operation is exercised.

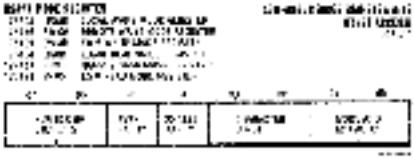
<02> RECEIVE ENABLE (REN)

0 = Disable
1 = Enable

<01> DATA TERMINAL READY (DTR)
For the remote line Data Terminal Ready (DTR) is asserted in the USART Command Register. The SVC sets DTR when the remote line is enabled by the switch on the system control panel. DTR enables the modem to receive a call and indicates that the USART is ready to accept data.

<00> TRANSMIT ENABLE (TXEN)

0 = Disable
1 = Enable



<07:06> STOP BITS

00 = Invalid
*01 = 1 Stop Bit
11 = 2 Stop Bits

<05> EVEN PARITY

<04> ENABLE PARITY

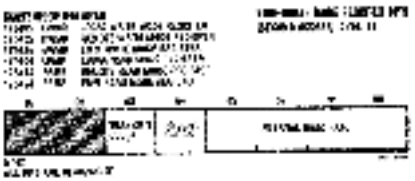
<03:02> CHARACTER LENGTH

00 = 5 Bits
01 = 6 Bits
*11 = 8 Bits

<01:00> BAUD RATE FACTOR

00 = Sync.1X
*01 = Async.1X
11 = Async.64X

NOTE: * = NORMALLY USED VALUE



<07:06> RESERVED

<05> TRANSMIT CLOCK
Selects the source clock for the transmit frequency where:

1 = Internal clock
0 = External clock

If set to 0, External clock, the external USART clock is used as transmit frequency. The baud rate is selected via bits <03:00>. The External baud rate must be the same for transmit and receive. If set to 1, Internal clock, the clock source is Internal (pin 9 of USART). The standard setting is Internal clock for CTY and E-ETHRN, External clock for RTY.

<04> RECEIVE CLOCK
Selects the source clock for the receive frequency where:

1 = Internal clock
0 = External clock

If set to 0, External clock, the External USART clock is used as receive frequency. The baud rate is selected via bits <03:00>. The External baud rate must be the same for transmit and receive. If set to 1, Internal clock, the clock source is internal (pin 9 of USART). The standard setting is Internal clock for CTY E-ETHRN, External clock for RTY.

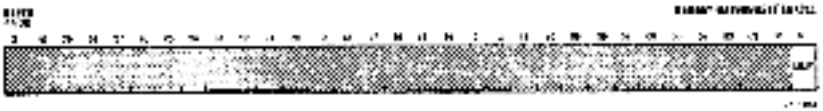
<03:00> INTERNAL BAUD RATE
If Internal baud rate is selected via bits <3> or <04>, bits <03:00> determine the frequency (baud rate).

Bits <03:00>	BAUD RATE
0000	50
0001	75
0010	110
0011	134.5
0100	150
0101	300
0110	600
0111	1200
1000	1800
1001	2000
1010	2400
1011	3600
1100	4800
1101	7200
1110	9600
1111	19.2K

The default setting for these bits are as follows:
CTY: 38(K)
STERM: 38(K)
CTY: 07(K)

NOTE

Each of the DUARTS on the console module has a 8-bit mode register, one for each port: local port (CTY), remote port (CTY) and the ENK port (STERM). Access to this register is achieved by addressing an 8-bit register twice. The first access addresses the low-order bits <07:00> and the second access addresses the high-order bits <15:08>.



Memory Management Control - The action of translating a virtual address to a physical address is governed by the setting of the MEMORY MAPPING ENABLE (MME) bit in the MAPEN internal processor register.

<31:01> RESERVED

<00> MME
When MME, the MEMORY MAPPING ENABLE BIT is set to 1, memory management is enabled. When MME is set to 0, memory management is disabled. At processor initialization time, MAPEN is initialized to 0.



<31:14> On a write to the PBnn (WTPR) these bits must be zero, but on a read (mfpr) they will be undefined.

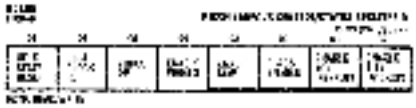
<13:12> DIAGNOSTIC WRITE LENGTH/STATUS <1:0> R
When the MBox is using the ABUS in diagnostic mode, the bits stored in these locations are substituted for the normal data length/status bits when writing to the I/O adapter register file. They are used for diagnostics only.

<11:08> DIAGNOSTIC WRITE COMMAND/MASK <3:0> R
When the MBox is using the ABUS in diagnostic mode, the bits stored in these locations are substituted for the normal command/mask bits when writing to the I/O adapter register file. They are used for diagnostics only.

<07:02> RESERVED

<01> SLOW ARRAY ACCESS MODE
When set, memory access time from the array is repeatable (and slow). This is for diagnostic reasons. The memory time is greater than the normal access time plus the refresh time.

<00> INHIBIT MEMORY OVERLAP
When set, memory overlap is inhibited (i.e., not starting another array read unless the data from the previous array read has been extracted from the array register file [DC109]). This is needed if the clock is to be stopped and continued. This is also used to do single step on DMA transfers by effecting DMA runs.



<07>	HOLD STATE RESET
CL09 HOLD STATE RESET H
Initialize the VAX CPU but do not clear error status information.

<06>	DIAG CLR PE
CL04 CLR PE H
Output of parity error F/F. Clocks PXCAS and PERRS registers during normal operation. When in the test bench, the T-11 may set CL09 DIAG TRIGGER, which may be used as a scope trigger by console diagnostics.

<05>	FORCE PARITY ERROR
CL09 FORCE PE H
Write bad parity in the T-11 RAM.

<04>	ENABLE PROM S1
CL09 ENA PROM S1
Select upper 4Kb of PROM.

<03>	ENABLE MAP
CL09 ENA MAP H
Enable virtual to physical address translation of the T-11 RAM address by paging (mapping) RAM.

<02>	FLAGS ENABLE
CL09 ENA CPU FLAGS H
Enable console (T-11) interrupt requests generated by the VAX CPU. Enable VAX CPU interrupt requests generated by the console.

<01>	ENABLE TOY INTERRUPT
CL09 ENA TOY INTR H
Enable T-11 interrupt requests generated by Time of Year Clock every 1 ms.

<00>	ENABLE T-11 INTERRUPT
CL09 ENA TOY INTR H
Enable T-11 interrupt requests to the T-11 program.



<0:7> RTERM INTERRUPT ENABLE
CLD9 TERM IE H
Enable C-11 interrupt request generated by remote line receiver ready condition or change in modem status.

<0:6> RTERM DSRS
CLD9 RTERM DSRS H
Enable 8600 remote line modems to operate at different (split) transmit and receive band rates.

<0:5> SIMULATE CPU ERROR
CLD9 SIM CPU ERROR H
Force C-11 interrupt request normally generated by VAX CPU control store parity error.

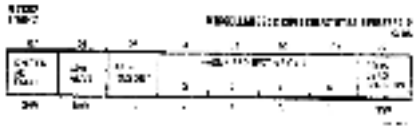
<0:4> CSM REQUEST ENABLE
CLD9 CSM REQ ENA E
Request that the VAX CPU microprogram in the FBox enable support microcode.

<0:3> CPU RESET
CLD9 CPU RESET
Initialize VAX CPU (master reset).

<0:2> MEMORY BUS ENABLE
CLD9 MBX BUS ENABLE E
Enable main memory array to respond to read/write requests by the MBox.

<0:1> ABUS ABORT INITIALIZE
CLD9 ABUS ADAPT INIT H
Initialize CBIA and ABus devices.

<0:0> ENABLE MAINTENANCE CLOCK
CLD9 ENA DMAINT CLK H
Enable alternate clock source (50Hz clock) for testing of DBus XBLY timeout counter.



<07> SYSTEM AC FAULT
CL09 SYS AC FAULT H
Latched AC LV indication from EBM. Bit may be written to simulate AC fail for test purposes.

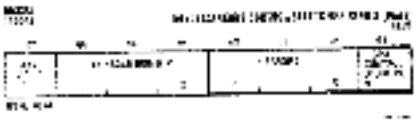
<06> CPU ALIVE
CL09 CPU ALIVE H
VAX CPU executing another instruction. Set by EBOX at IRD. Bit is cleared and rechecked periodically to determine that VAX CPU program is running.

<05> REPLY TIMEOUT
CL30 REPLY TIMEOUT H
REPLY was not received on CBus during CBus read/write operation. I-11 program is interrupted.

<04:01> -ABUS REQUEST <3:0> H
-CL09 ABUS REQ H <03:00>
VAX CPU restart requests from system's four possible ABus adapters. A restart request originates from another (CI connected) processor. Bits are asserted when in the zero state.

<00> ABUS DEAD INTERRUPT
CL09 ABUS DEAD INTERRUPT H
Latched VAX CPU restart request from one or more of the system's four possible ABus adapters (such as the OR of bits <04:01>). The console T-11 <01> diagram is interrupted. Bits may be written to simulate ABUS DEAD for test purposes.

VAX 8600/8650 REGISTER DESCRIPTION
MCSR3 (SSR0)



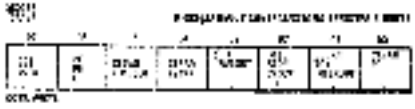
<07> -CPU AC LO H
-BNM3 CPU AC LO H
AC LO is originated from the BNM. Indicates that AC power has dropped, but not enough to cause loss of DC power. DC LO, BNM3, indicates that software operation is unpredictable.

<06:04> CONSOLE IDENTITY <0:2>
CI(0) is <0:2> H
Console identity bits programmed at backplane to establish access privileges on CNA multi-drop bus. These bits originate from backplane jumpers (see CLIX and drawing BD-10A01-0-SEC). Currently these bits are unused and read as 0's (jumpers out).

<03:01> -ERROR <2:0> H
-CI(0) ERROR <2:0> D
Complement of error code from CMAX specifying type of control store error condition in the VAX CPU. These signals originate on BNM3 and are transmitted to the console module (print CL(9)), used by the console for VAX CPU CS error correction and reporting.

0-MBOX CS PE	3-FBOX DRAM PE	6-MBOX CS PE
1-FBOX MULTIPLIER CS PE	4-IBOX DRAM PE	7-NO ERROR
2-FBOX ADDER CS PE	5-IBOX CS PE	

<00> -CPU CONTROL STORE PARITY ERROR H
CI(9) CPU CS PE L
This bit indicates that a CPU control store parity exists and will interrupt the console program at vector 110. This signal is asserted if any of the control store parity errors specified by bits <3:0> exist, or if STIMULATE CPU ERROR, MCSR bit <05> is set. The latter is for diagnostic purposes only.



A write to MCSR1 enables a decode of DAL<07:05> to provide 8 diagnostic functions.

<07:05> FUNCTION
The following functions/operations are performed when the bits in this field equal an octal 7, 6, 5, 4, 3, 2, 1, or 0. Note: Only one function can be performed at a time.

7 SET TIMEOUT L
Set Reply Timeout (MCSR2<6>), simulates the console reboot request by a VAX CPU. Generates a T-11 interrupt request.

6 SET PARITY ERROR L
Sets the parity error indicator to simulate detection of bad parity in the I-11 RAM. Generates a T-11 interrupt request.

5 CLEAR TIMEOUT L
Clears the Reply Timeout (MCSR2<6>) and the corresponding T-11 interrupt request.

4 CLEAR TEST L
Clears TEST (CMISC<6>) and the corresponding T-11 interrupt request.

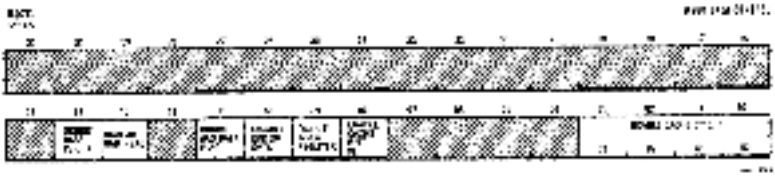
3 SET TIMEOUT L
Set the Reply Timeout (MCSR2 <6>). Simulates a Reply Timeout condition on the QBus. Generates a T-11 interrupt request.

2 Set STOP CLOCK L
Stop generation of SDB clocks.

1 CLEAR TOY INTERRUPT L
Clear the T-11 interrupt request generated by the Time Of Year Clock.

0 CLEAR PARITY ERROR L
Clear the T-11 interrupt request generated by the T-11 RAM parity error.

<04:00> RESERVED



<31:15> On a write to the MBox (MTPR) these bits must be zero, but on a read (MFPR) they will be undefined.

<14> INHIBIT DATA PARITY ERROR CHECK
When Set, MD BUS WRITE PARITY ERR and CACHE DATA PARITY ERR errors will not be detected or reported by the MBox.

<13> DIAGNOSTIC CHECK RAM READ
When set, check bits will be read from the selected Cache RAMs. This function will only be used by diagnostics.

<12> RESERVED

<11> INHABIT BAD DATA FLAG
When set, will inhibit the bad data code check bits being generated. See MDECC for a description of the BAD DATA CODE.

<10> DISABLE ECC ON DATA
When set, ensures that no correction takes place on the data. Error information will still be logged and reported unless INHIBIT ERROR REPORTING bits are set in register MERR.

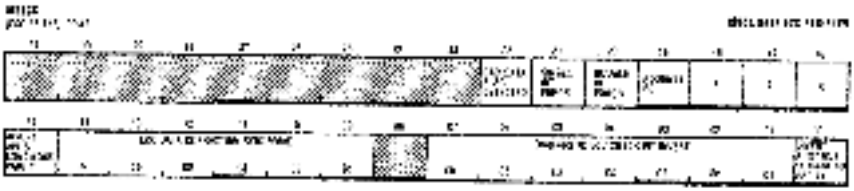
<09> ENABLE INVERT REGISTER
When set, allows the data stored in the DATA CHECK BIT INVERT REGISTER to complement the check bits generated on any write data. The check bit invert bits are in register MDECC. Also allows bad ABus parity to be generated for DMA array reads if MDECC <0> had been written.

<08> ENABLE CACHE BYTE PARITY ERROR
When set, allows bad byte parity to be written into Cache from a CP request as determined by CACHE PERR <03:00> bits. Once this bit is enabled, the MBox will invert parity for one MBox request. After the first request the bit will stay set but no more parity inversions will take place. To enable a second parity inversion the ENA CACHE BYTE PE bit must be reset, then set.

To invert byte parity in Cache, the ENA CACHE BYTE PERR bit must be set along with CHA CACHE PERR <03:00> and IGN DAT PERR CHK. If the IGN DAT PERR CHK is not set the MBox will detect parity errors on the incoming data. This function can be used to produce WRITE DATA PARITY ERRORS, REGISTER WRITE PARITY ERROR and ABUS DATA UP PARITY ERRORS on a CP write.

<07:04> RESERVED

<03:00> ENABLE CACHE BYTE PARITY ERROR
When set in conjunction with ENABLE CACHE BYTE PE, will generate wrong parity on byte <03:00> being written into the Cache during CPU longword writes. This is also used to generate single longword failures for CPU ABus write.



MDECC (MBox Data ECC Status Register) - This register is used in conjunction with MBox registers 70 (byte 2), 60 (byte 1), and 50 (byte 0).

<31:24> Reserved

<23:16> Source: MCDM (ECC) MBox Reg 70 (DATA ECC ERROR)
Hold xfr ECC4 ERR HLD and 11b

<25> Reserved

<22> BAD DATA SYNDROME DETECTED
ECC4 BD ERR (MCDM)
The bad data bit is added into the ECC check bit generation whenever "broken" bad data is written into either the cache or the array. A read to that location will result in the detection of a bad data code and cause this bit to be set. Note that to read check bits from Cache a cache byte parity error must have been taken to invoke the check bit read.

<21> SINGLE BIT ERROR
ECC1 SB ERR (MCDM)
The data word read from cache or main memory had a single bit (ECC correctable) error.

<20> DOUBLE BIT ERROR
ECC4 DB ERR (MCDM)
The data word read from cache or main memory had a double bit error or a detectable multiple bit error.

<19> ADDRESS PE
ECC4 AP ERR (MCDM)
The word fetched from memory had either written or read from the wrong location. The ECC code indicates that the parity of the address written and the parity of the address read from are different. ECC is generated over the data bits and a parity bit computed over the physical address bits <29:0>.

<18:16> 110
ECC2 DATA <33:32> (MCDM)
These bits should always return as a binary 110. Otherwise, this field position can be used as a key to confirm the location of MDECC in a Stack Frame.

<15:08> Source: MCDM (ECC) MBox Reg 60 (DATA SYNDROME)
Hold xfr ECC4 ERR HLD

<15> INVERT ADDR LONGWORD PARITY
ECC3 FMP INVERT ADG (MCDM)
When this bit is set, and bit <01> in Register 10 (Data Control) is set the ECC MCA will generate bad Addr longword parity.



<14:00> ECC DATA CORRECTION SYNDROME
ECC) SYN REG <32:1> (MCDM)
ECC) SYN REG <32:1> (MCDM) = This field corresponds to the syndrome generated by the XMI chip and indicates the failing bit number.

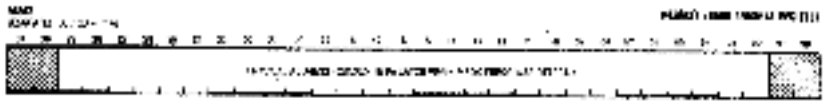
SYNDROME	(MSB)	10	0421000 8888885555554444433333222211111
ID OCTAL	(LSB)	07	0000421 6543210765432165432165432165321
DATA DTY	(MSB)	AA	CCCCCCC 3333333222221111111100000000
	(LSB)	3F	7654321 0987654321098765432109876543210

<08> Unused

<07:00> Source: MCDM (ECC) MBox Reg 50 (DATA CHECK INVERT)

<07:01> DIAGNOSTIC CHECK BIT INVERT <CP:C1>
MCD3 <CP:C1> INVERT REG (MCDM)
When set the corresponding ECC check bits will be inverted.

<00> INVERT ARRY BIB LONGWORD PARITY
MCD3 INV INVERT REG (MCDM)
When this bit is set, and bit <01> in Register ID (Data Control 3) is set, then the longword parity generated on ARRAY Adr data will be inverted when the ECC MUX is the check mode. This bit is write only and is read via bit <1> in MUXIC (MBox Reg.ster 50).



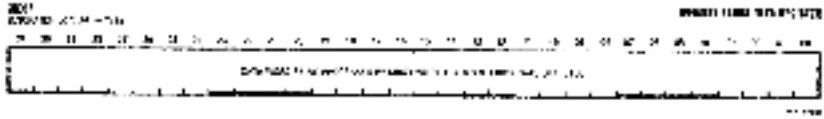
This is a copy of MAPL/3 (ADA/ADB) MBOX REG 7C (ERROR ADDR). It contains the physical address present at the output of the DA Mux when the MBox detected an error. Interpreting this address depends on the MBox Data Destination Code (MSTAT1 <31:30>) and the MBox Cycle Type (MSTAT1 <29:26>).

1. CP Initiated (Non I/O) Read/Write Operations - MBAR will contain the address of the longword that was associated with the error unless:

   CP Refill or WRITEBACK - (The CP request caused either a Cache Refill Cycle, or a Cache Writeback Cycle). Then MSTAT1 <25:24> (Longword Count <A3:A2>) must be used to determine the address of the longword that was associated with the error. If MSTAT1 <25:24> equal MBAR <01:00> then MBAR contains the address of the longword associated with the error. Otherwise, you must substitute MBAR <03:02> with MSTAT <25:24> to determine the address.

2. CP Initiated I/O Read/Write Cycles - MBAR is not valid for any CP initiated I/O Cycles.

3. ABus Initiated Read/Write Cycles - MBAR will contain the starting address of the data to be processed (i.e., longword, quadword, or octaword). MSTAT1 <25:24> (Longword Count <A3:A2>) must be used to determine the address of the longword that was associated with the error. If MSTAT1 <25:24> equal MBAR <03:02> then MBAR contains the address of the longword associated with the error. Otherwise, you must substitute MBAR <03:02> with MSTAT <25:24> to determine the address.



This is a copy of MCTL/3 (MDF) MBOX REG 7E (DATA ERROR). It is only valid (held) for ABus Parity Errors (DMA Data Pin and CPU Read Pen only), ABus BDC Errors, and CPU Write PEs. It contains the data word latched in the MIO Mux when the error occurred.



If these bits are reset, they only prevent the status bits from being latched; they do not prevent the error from being reported, the trap still occurs. If these bits are reset, software will see what appears to be spurious traps or interrupts if the corresponding error is detected.

<31:24>
On a write to the MBox (MTPR) these bits must be zero, but on a read (MFPR) they will be undefined.

<23:16> MBOX STATUS REPORT ENABLE
Enable value = FF HEX.

<23> CPR B PARITY ERROR

<22> CPR A PARITY ERROR

<21> ABUS DATA PARITY ERROR

<20> ABUS CONTROL PARITY ERROR

<19> ABUS ADDRESS PARITY ERROR

<15:12> RESERVED

<11:08> MBOX TB ERROR REPORT ENABLE

<11> VALID PARITY ERROR

<10> PTE B PARITY ERROR

<09> PTE A PARITY ERROR

<08> TAG PARITY ERROR

<07:00> MBOX DATA ERROR REPORT ENABLE

<07> WB BYTE 3 PARITY ERROR

<06> WB BYTE 2 PARITY ERROR

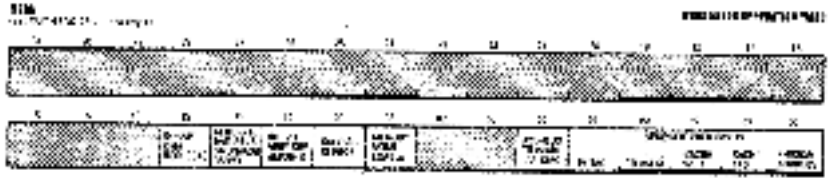
<05> WB BYTE 1 PARITY ERROR

<04> WB BYTE 0 PARITY ERROR

<03> CACHE DATA PARITY ERROR

<02:01> RESERVED

<00> CACHE BYTE ACTIVE PARITY ERROR



MERR (MBox Error Generation Register) - This register is made up of MBox registers 10 (byte 1) and 14 (byte 0).

<31:26> Reserved

<25:00> Source: MCCI (CRB) MBox Reg 10 (MCC CONTROL 3)
Notes: This field is set by loading the Control Register on CRB4.

<15:11> Reserved

<10> INHIBIT DMA REQUESTS
MCC1 INH DMA REQ
When set, the MBox will not honor any DMA requests.

<1> GENERATE BAD ABUS CMD/MASK PARITY
MCC1 GEN BAD PAR
When set, ABus Command/Length and ABus Mask/Status will be transferred with even parity.

<10> INHIBIT ARRY UNC REPORTS
MCC1 INH ARY COR# REP
When set, this bit prevents the MBox from reporting MCC correctable errors. The errors will still be corrected as usual.

<09> IOA DIAG IN PROG
MCC2 IOA DIAG IN PROG
When set, the ABus command/mask field and length/status lines are driven from MCC CONTROL REGISTER 1C <03:00>.

<08> MEMORY MGMT ENABLE
MCC1 MEM MAN EN
When set, memory management is enabled. All virtual references are then translated by the TB. When this bit is cleared, all references will be treated as physical and all TB parity error checking will be disabled.

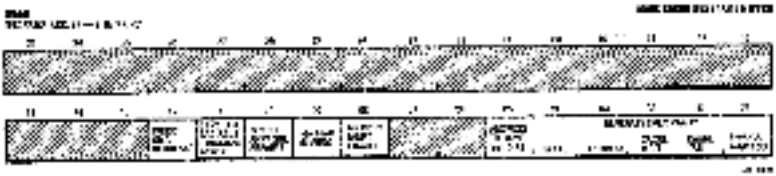
<07:00> Source: MAPP (REG) MBox Reg 14 (ADDR CONTROL 2)

<07:06> Reserved

<05> ADDRESS TB RAM VIA DIAG
MAPP ADR RAM DIAG
The effect of this bit depends on the cycle type:

Write TB = Causes the TB RAMs containing physical address bits <12:09> and selected by the IVA lines to be written instead of those selected by the EVA lines. Used for diagnostic testing of TB RAMs.

Read TB = Causes the contents of the TB RAMs containing physical address bits <12:09> and selected by the IVA to be read instead of those selected by the EVA lines. Used for diagnostic testing of TB RAMs.



Diagnostic Read Cache Tag Address - Causes the written bit, written parity valid bits and cache tag parity to be read in place of the cache tag address bits. Used for diagnostic testing or the cache tag RAMs.

<04:00> GENERATE EVEN PARITY
When set, the bits in this field will cause the MBox to generate even (bad) parity for the corresponding function.

<04> GENERATE EVEN PARITY TO TAG
MCGT EPM TO TAG PAR
When set, a write to the TB will result in the TB tag being written with even parity. A read to this TB location will result in a TB TAG parity error.

<03> GENERATE EVEN PARITY TB VALID
MAPT GEN TB VAL ERR
When set, a write to the TB will result in the TB Valid bit field being written with even parity. A read to this TB location will result in a TB Valid parity error.

<02> GENERATE EVEN PARITY CACHE WBIT
MAPT GEN EVN WBIT PAR
When set, parity for the cache written bit will be complemented before being written in the cache. A read to this cache location will result in a cache WBit parity error.

<01> GENERATE EVEN PARITY CACHE TAG
MAPT GEN EVN TAG PAR
When set, the cache data address tag is stored with even parity during a cache write. A read to this cache location will result in a Cache Tag parity error.

<00> GENERATE EVEN PARITY PHYSICAL ADDRESS
GEN EVN PA PAR
The effect of this bit depends on the type of operation the MBox is performing.

Cache or Array Writes - When set, the address parity bit generated as part of the ECC character generation will be complemented. If the write was to the array this condition will be detected when that array location is next read. If the write was to the Cache a Cache Byte parity error will have to be forced by some other means in order to detect this condition.

DMA - The MBox will detect an Abus Address Parity Error.

CP to I/O Transfers - The Abus Adapter will detect bad address parity and set CP I/O Buffer Error.

IPR Writes - Bad parity will be forced in both PTE A and PTE B.



This register is made up of the following NBOX REGISTER 2C: 2C (byte 3), 2B (byte 2), 2A (byte 1), and 29 (byte 0).

<31:24> ERROR REGISTER 2C (MCC STATUS 1)
SOURCE: MCCJ (CSR) NBOX REG 2C (MCC STATUS 1)
ACCESS: Read only
HELD AT: MCCJ CYC ERR + T2

<31:30> NBOX DESTINATION CODE <1:0>
MCLK (ANY CLOCK) UP <1:0>
Indicates the destination code associated with the error.

Code	Destination
00	IBUF (IBox - Don't Load Tail Pointer)
01	IMD (IBox - FETCH/STORE Operand)
10	PMD (PBox - FETCH/STORE)
11	IBUP (IBox - PBTCH - Load Tail Pointer)

<29:26> NBOX CYCLE TYPE
MCCJ N CYC TYP <3:0>
Indicates the microword cycle type associated with the error.

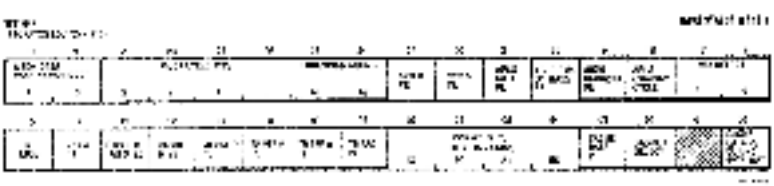
CODE	CYCLE	CODE	CYCLE
0000	NOP	1000	ABUS
0001	READ SEQ	1001	CP RSFILL
0010	ACTIVE CAS	1010	UNVAL TB
0011	WRITEBACK	1011	TB CYC
0100	ABUS ARRAY WRT	1100	CP ARRAY WRT
0101	DATA CORRECTION	1101	CP WRITE
0110	CLEAR CACHE	1110	CP READ
0111	TB PROBE	1111	ABUS RSFILL

<25:24> LONGWORD COUNT
MCCJ WD CNT <01:02>
Indicates the longword that was being processed when the error was detected.

<23:16> NBOX REGISTER 2B (MCC STATUS 3)
SOURCE: NBox Register 70 (MCA: FRA) MCCJ (CSR) MBOX REG 2B (MCC STATUS 3)
ACCESS: Read Write
HELD AT: MCCK CYL NBX NBN + T0

<23> CYCLE PARAMETER RAM B PARITY ERROR
MCCC CPR PERR B
Indicates that a Cycle Parameter RAM B parity error was detected. The NBox response is unpredictable.

<22> CYCLE PARAMETER RAM A PARITY ERROR
MCCC CPR PERR A
Indicates that a Cycle Parameter RAM A parity error was detected. The NBox response is unpredictable.



<21> ABUS DATA PARITY ERROR
MCD3 ABUS DAT PERR
Indicates that a longword parity error was detected on the ABus Data Field during either a CP I/O READ or DMA WRITE Cycle.

<20> ABUS COMMAND OR MASK PARITY ERROR
MCD4 ABUS CMD PERR
If bit <18> is set, this bit indicates that a parity error was detected on the Command/Length Field during an ABUS Command Address cycle. If bit <18> is reset then this bit indicates that a parity error was detected on the Mask/Status Field during an ABus Data Cycle.

<19> ABUS ADDRESS PARITY ERROR
MAP2 ABUS ADR PERR
Indicates that a parity error was detected on the Address Field associated with an ABus Command Address Cycle.

<18> ABUS COMMAND/ADDRESS CYCLE
MCCJ ABUS ID CMD
Indicates that the MBox was executing a DMA Command Address cycle when an error was detected.

<17:16> IOA SELECT <01:00>
MCC4 ABUS SEL <01:00>
Indicates the ABus Adapter that was selected when the error was detected.

<15:08> MBOX REGISTER 24 (ADDRESS STATUS)
SOURCE: MAPP [REG] MBOX REG 24 (ADDR STATUS)
ACCESS: Read Only
HOLD AT: Self holding at 0

<15> TB MISS
MAP5 TB MIS
Indicates that the TB TAG (indexed by VA <16:09>) did not match VA <30:17> or the valid bit was not set. The MBox will send a Fork Status of "A" (TB Miss) back to the CBox.

<14> BLOCK HIT
MAP4 SHP BLOCK HIT
Indicates that either the Tag for Cache 0 or the Tag for Cache 1 matched PA <28:13>, at least one valid bit was set, and an I/O adapter was not selected.

<13> CACHE 0 TAG MISS
MAP3 C0 TAG MAT
Indicates that Tag in Cache 0 did not match PA <28:13>.

<12> CACHE MISS
MAPL HOP CACHE HIT
Indicates that either PA <29:13> failed to match both the Cache 0 Tag and the Cache 1 Tag or, if a match did occur then the valid bit for the tagged cache(s) was not set.

<11> IB VALID PARITY ERROR
MAPP IB VAL ERR
Indicates that a parity error was detected on the valid bit when the IB was read. The MBox will send a Port Status of "6" (IB Error) back to the EBox.

<10> IB PTE B PARITY ERROR
MAP3 PTE B PAR ERR
Indicates that a parity error was detected when the IB PTE RAM containing PA <29:23> was read. The MBox will send a Port Status of "6" (IB Error) back to the EBox.

<09> IB PTE A PARITY ERROR
MAP4 PTE A PAR ERR
Indicates that a parity error was detected when the IB PTE RAM containing PA <29:23>, PROTECTION <3:A>, and MODIFY bits were read. The MBox will send a Port Status of "6" (IB Error) back to the EBox.

<08> IB TAG PARITY ERROR
MAP3 IB TAG PAR ERR
Indicates that a parity error was detected when the IB TAG RAMs were read. The MBox will send a Port Status of "6" (IB Error) back to the EBox.

<07:00> MCDU REGISTER 20
SOURCES: MCDU ASH 20 (DATA STATUS)
ACCESS: Read Only
ERLG AT: MCDU VOP CYC ERR - T3

<07:04> CPU WRITE (BYTE IN ERROR)
UP05 WB BYT <3:0> PERR (MCDU)
This field indicates that a parity error was detected on the data received from the CPU during a CPU write. Furthermore, this field indicates which byte(s) had the parity error.

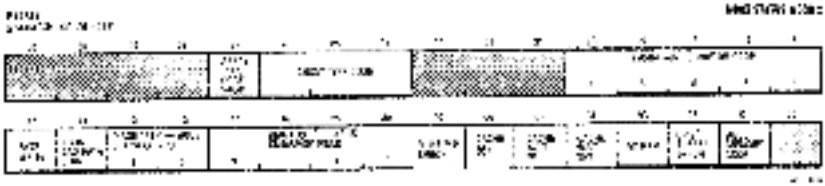
<03> CACHE DATA PARITY ERROR
UP05 CACHE BYT PERR (MCDU)
Indicates that a byte parity error was detected on data read from cache during any Cache operation other than a CPU Byte Write WB. Includes: CPU Read, DMA Read, Writeback, and DMA Masked Write.

<02> CACHE 1 SELECT
MAPL CACHE 1 SAT
This bit indicates the Cache that was selected when the error was detected.

<01> ARRAY READ
MCDU ANY REFILL
Indicates that a Cache Refill was in progress when an error was detected.

<00> CACHE DATA PARITY ERROR DURING BYTE WRITE
UP05 CACHE BWRT PERR (MCDU)
Indicates that a Cache Data Parity Error was detected during a CPU Byte-write Operation.

MSTAT2



This register is made up from the following MBox Registers: 54 (byte 2), 5C (byte 1), and 58 (byte 0).

<31:24> VMS SUPPLIED
This field is supplied by VMS and describes the array type selected at the time that the error occurred.

<23:28> RESERVED

<27> ARRAY TYPE CODE VALID
Indicates that the Array Type Code in bits <26:24> is valid.

<26:24> ARRAY TYPE CODE
Indicates the Array type that was selected when the error occurred.

CODE	ARRAY TYPE	CODE	ARRAY TYPE
000	Reserved	100	Reserved
001	16 Mb Array	101	Reserved
010	04 Mb Array	110	Reserved
011	64 Mb Array	111	Reserved

<23:21> RESERVED

<20:16> MBOX REGISTER 54
SOURCE: MAP4 (RAW) MBOX REG 34 (PAMM)

<20:16> PAMM CONFIGURATION CODE <all>
MAP4 PAMM CODE <all>
The Error Handling Microcode uses VSAR <29:22> to address the PAMM and then loads the five-bit PAMM code into this field. Normally this field will indicate the Array Module or I/O Adapter selected when the error was detected. If, however, the error involved a CP to I/O transfer, then this field will not be valid.

CODE	SELECTS	CODE	SELECTS
00	Array Slot 0	18	I/O Adapter 0
01	Array Slot 1	19	I/O Adapter 1
02	Array Slot 2	1A	I/O Adapter 2 (Not used)
03	Array Slot 3	1B	I/O Adapter 3 (Not used)
04	Array Slot 4	1C	Reserved
05	Array Slot 5	1D	Reserved
06	Array Slot 6	1E	Reserved
07	Array Slot 7	1F	Non-existent Address

08 - 17 Reserved.

<15:08> MBOX REGISTER 5C
SOURCE: RCLJ (CRA) MBOX REG 5C (NOC STATUS 2)

<15> SYTB WRITE
NCC7 OR SYTB WRITE
Indicates that the MBox was servicing a byte write request.

<14> ABUS BAD DATA CODE
MCC4 AB BAD DAT ERR
Indicates that the CP IO read data received from the ABus Adapter was marked as bad data via the ABus Status Field.

<13:12> DIAGNOSTIC ABUS LENGTH/STATUS FIELD
MCC4 LABUS STAT <1:0>
Indicates the state of the ABus length and Status lines during an I/O diagnostic operation.

<11:00> DIAGNOSTIC ABUS COMMAND/MASK FIELD
MCC4 LABUS MSK <3:0>
Indicates the state of the ABus Command and Mask lines during an I/O diagnostic operation.

<07:00> MBOX REGISTER 50
SOURCE: MCCJ (CPB) MBOX REG 50 (MCC STATUS 4)

<07> MULTIPLE ERROR
CBB5 MULT ERR (MCCJ)
Indicates that a second MBox error was detected before the MBox could read and clear the first error. Most or all of the information associated with the second error will be lost.

<06> CACHE TAG PARITY ERROR
CBB5 CACHE TAG PERR (MCCJ)
Indicates that a parity error was detected in the address and valid bit portion of the Cache tag.

<05> CACHE WRITTEN BIT PARITY ERROR
CBB5 TAG WRT PERR (MCCJ)
Indicates that a parity error was detected in the Cache tag Written Bit field. The MBox handles the request as though the written bit was a one.

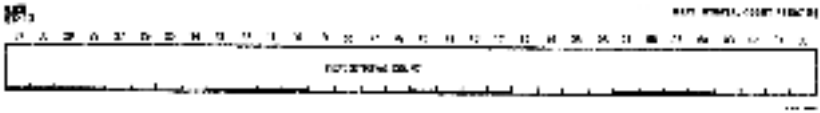
<04> CACHE WRITTEN BIT SET
CBB5 WRITTEN (MCCJ)
Indicates that the cache written bit was set when an error was detected.

<03> NON EXISTENT MEMORY
CBB5 NXM (MCCJ)
Indicates that either the memory request was to Non-Existent Memory or an I/O adapter attempted to address another I/O adapter. An MBox Fatal Error status code is sent to the CBox for CP initiated requests, and the DMA ERROR line is asserted to the I/O Adapter for I/O initiated requests. All writes are cancelled.

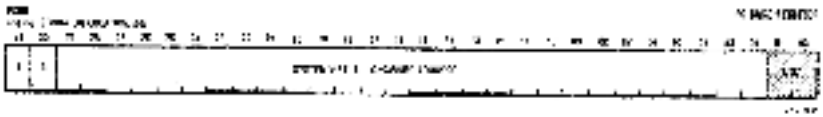
<02> CP I/O BUFFER ERROR
CBB5 CP IO BUF ERR (MCCJ)
Indicates that the selected ABus Adapter detected an error while processing a CPU request.

<01> ABUS MEMORY LOCK
CBB5 LOCK LINE (MCCJ)
When set, indicates that the Memory Lock line was being driven on the ABus. This line may be driven by either the MBox or an ABus Adapter.

<00> RESERVED

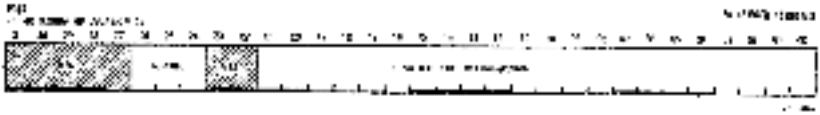


<31:00> NEXT INTERVAL COUNT
The Nxtcnt register is a write only register that holds the value to be loaded into ICR when it overflows. The value is retained when ICR is loaded. NICR is capable of being loaded regardless of the current values of ICR and ICCS.



<31:02> P0 BASE REGISTER
Contains the system virtual longword address of the start of Process 0 Page Table (P0VT).

<01:00> MBZ
Must be zero.

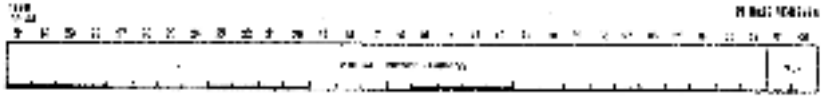


<31:27> MBZ
Must be zero.

<26:24> RESERVED

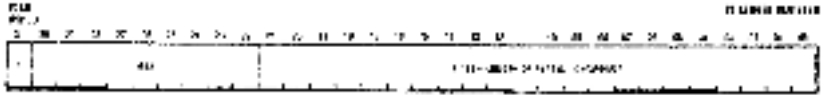
<23:22> MBZ
Must be zero.

<21:00> PROCESS 0 LENGTH REGISTER
Contains the size (in longwords) of Process 0 Page Table (P0PT).



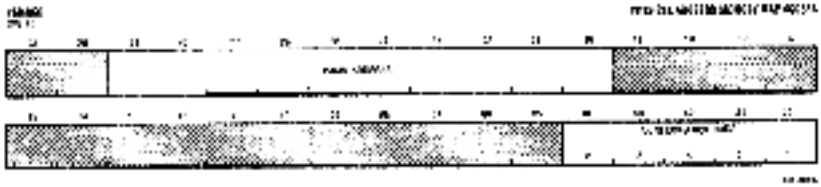
<31:02> P1BR (PROCESS 1 BASE REGISTER)
Base register for page table describing process virtual addresses from 2**31-1.

<01:00> MBZ
Must be zero.



<31:22> MBZ
Must be zero.

<21:00> P1LR (PROCESS 1 LENGTH REGISTER)
Length register for page table located by P1BR. Describes effective length of page table.



NOTE:

PAMACC and PAMDAT are used to write/read the PAMM. When writing the PAMM, the PAMM address and data is written to PAMACC with a MTPR. XMox ulxo code will carry out the write to the PAMM with an MBox register write. When reading, PAMDAC will be loaded with a MTPR PAMDAC, with the PAMM address to be read. The PAMM data will be gated from the PAMM to the PAMACC via MFPR PAMACC which will cause the EBox microcode to read the PAMM with an MBox register read.

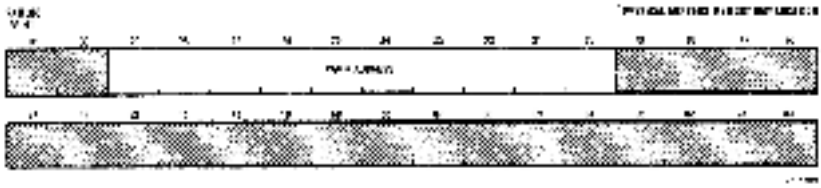
<31:30> RESERVED

<29:20> PAMM ADDRESS
The PAMM address specifies the PAMM address to be read or written. It can specify one of 1024 possible entries in the PAMM. Each PAMM address, when mapped corresponds to a 1-MByte window of physical address space. The bit alignment in the PAMM address field corresponds to the bit alignment of the physical address.

<19:05> RESERVED

<04:00> CONFIGURATION CODE
CODE A.0,4.2.1 - CONF CODE <4:1> defines Memory Array Card, Adapter Code, or XMM. (CONFIG CODE "A" set to one specifies on one mega-byte boundaries to inhibit writing data to Cache (used to inh bit writing I/O data to cache).

CONFIG CODE	SELECTS
00	Internal Array Slot 0 (Array Card 0)
01	Internal Array Slot 1
02	Internal Array Slot 2
03	Internal Array Slot 3
04	Internal Array Slot 4
05	Internal Array Slot 5
06	Internal Array Slot 6
07	Internal Array Slot 7 (Array Card 7)
08-17	Unused Codes
18	I/O Adapter 0
19	I/O Adapter 1
1A	I/O Adapter 2
1B	I/O Adapter 3
1C-1E	Unused Codes
1F	Non Existant Address



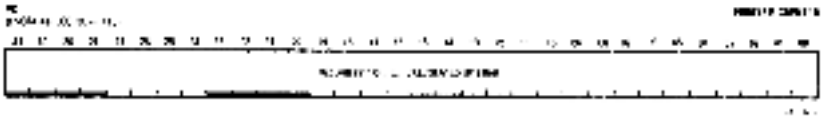
NOTE

See DAMADC. The PAMM is addressed on 1 Mega-byte boundaries, which is why the PAMM address starts at bit 20.

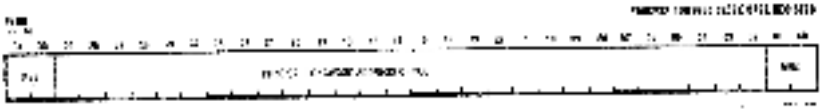
<31:30> RESERVED

<29:20> PAMM ADDRESS

<19:00> RESERVED



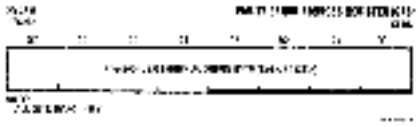
This register contains the PC to be used by VMS if it determines that an REI is possible. The contents of this register are determined by the Error Handling Microcode. Depending on the instruction in the pipeline that was associated with the error, this register will reflect the CPC, the ISA, or the ESA.



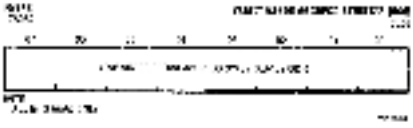
<31:30> MBZ
Must be zero.

<29:02> PHYSICAL LONGWORD ADDRESS OF PCB
The Process Control Block Base (PCBB) Register points to the process control block for the currently executing process. The PCBB Register is an internal privileged register, which contains the physical longword address of the Process Control Block (PCB). The PCB itself contains all of the switchable process context collected into a compact form for ease of movement to and from the privileged internal registers.

<01:00> MBZ
Must be zero.



NOTE: See description of PBIAS register.

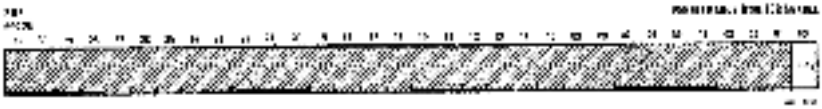


When a parity error is detected, the failing RAM address is stored in two 8-bit registers that may be read by the I-11. The Parity Error Bus (PBIAS) register stores the high-order address byte. The Parity Error RAM (PBRAS) register stores the low-order address byte.

In pass mode, the failing I-11 address (SAL<15:08> and RAS<7:00>) is stored in these registers.

In map mode, the failing virtual address (SAL<15:09> and RAS<7:00>) is stored. The physical address demands on the corresponding physical page number in the mapping RAM. The I-11 program must therefore calculate the physical address in order to determine which RAM location is bad.

In either case the console prints an error message and awaits rebooting either by the operator or by the operating system. This is done by the PROM code.



<3:0> RESERVED

<00> PERFORMANCE MONITOR ENABLE
Performance Monitor Enable controls a signal visible to an external hardware performance monitor. This bit is set to identify those processes for which monitoring is desired and to permit their behavior to be observed without interference from other system activity.

Writing a 1 to bit <00> via an MTPR instruction will provide an SCL logic "high" output on pin A09A07 of the CPU backplane (3B851).



<31> COMPATIBILITY MODE
When set, indicates that the processor is in PDP-11 compatibility mode. When clear, indicates that the processor is in native (VAX) mode.

<30> TRACE PENDING
This bit works in conjunction with Enable Trace bit <04>. If Enable Trace is set at the beginning of an instruction then the processor will automatically set this bit and thus cause a trap to the Trace Handler Routine.

The Trace Handler Routine will gather the desired information about the state of the system and REI (leaving the Enable Trace and Trace Pending bits alone). This will allow the next instruction to be traced. When the Trace Handler Routine wants to discontinue tracing, it will clear both bits (Enable Trace and Trace Pending).

<29:28> RESERVED

<27> FIRST PART DONE
This bit is set by long instructions that can be interrupted during execution (e.g., Move string). The REI following the interrupt will continue the interrupted instruction.

<26> INTERRUPT STACK
When set, the processor is executing on the interrupt stack. Any mechanism that sets this bit also clears current mode and previous mode (PSL bcurr 0). If an REI attempts to restore a PSL with IS=1 and non-zero current mode or zero IPL, a reserved operand fault is taken. When this bit is clear, the processor is executing on the stack specified by current mode. At bootstrap time, IS is set.

<25:24> CURRENT ACCESS MODE
This field indicates the access mode of the currently executing process, as follows: 0 = KERNEL, 1 = EXECUTIVE, 2 = SUPERVISOR, 3 = USER

<23:22> PREVIOUS ACCESS MODE
This field is loaded from the Current Access Mode by exceptions and CHMx instructions. It is cleared by interrupts, and restored by REI's.

<21> RESERVED

<20:16> INTERRUPT PRIORITY LEVEL
Indicates the current processor priority, in the range 0 to 1F (Hex). The processor will accept interrupts only at levels greater than the current level. At bootstrap time, IPL is initialized to 1F (Hex).

<15:08> RESERVED



<07> DECIMAL OVERFLOW TRAP ENABLE
When this bit is set, it forces a decimal overflow trap after execution of an instruction that had a conversion error, or produced a result with a decimal overflow (e.g., numeric string, or packed decimal). When this bit is clear, no trap will occur, however, the condition code V bit will still set.

<06> FLOATING UNDERFLOW EXCEPTION ENABLE
When this bit is set, it forces a floating underflow exception after execution of the instruction that produced a result with an underflow (e.g., a result exponent, after normalization and rounding, less than the smallest representable exponent for the data type). When this bit is clear, no exception occurs.

<05> INTEGER OVERFLOW TRAP ENABLE
When this bit is set, it forces an integer overflow trap after execution of an instruction that produced an integer result that overflowed or had a conversion error. When this bit is clear, no integer overflow trap will occur, however, the condition code V bit will still set.

<04> TRACE ENABLE
When this bit is set at the beginning of an instruction, it will cause trace pending <30> to set. When trace pending is set at the end of an instruction, a trace fault is taken before the execution of the next instruction. When TP is clear, no trace exception occurs.

<03:00> CONDITION CODES: N, Z, V, C

N BIT - When set, the N (negative) condition code bit indicates that the last instruction that affected this bit produced a negative result. If this bit is clear, the result was positive or zero.

Z BIT - When set, the Z (zero) condition code indicates that the last instruction which affected this bit produced a result which was zero. When this bit is clear, the result was non-zero.

V BIT - When set, the V (overflow) condition code bit indicates that the last instruction which affected this bit either had a conversion error or produced a result whose magnitude was too large to be properly represented in the operand which received the result. When this bit is clear, there was no conversion error or overflow.

C BIT - When set, the C (carry) condition code bit indicates that the last instruction which affected this bit either had a carry out of the most significant bit of the result or a borrow into the most significant bit. When this bit is clear, there was no carry or borrow.



<31> VALID BIT
Governs the validity of the M bit and PFN field. V=1 for valid; V=0 for not valid. When V=0, now M and PFN fields are reserved for DIGITAL software.

<30:27> PROTECTION FIELD
This field is always valid and is used by the CPU hardware even when V=0.

<26> MODIFY BIT
When the Valid bit is clear, M is not used by CPU hardware, and is reserved for DIGITAL software and I/O devices. When the Valid bit is set, M shows whether the page has been modified. If M is clear, the page has not been modified. If M is set, the page may have been modified.

M is cleared only by software. It is set by EBox Microcode (firmware). In addition, it may be set by the probe-write instruction (PROBEW) or by an implied probe-write. M is not set if a fault occurs in an instruction which would otherwise have modified the page.

For example, if a write reference crosses a page boundary where the first page is not accessible and the second page is accessible, the reference will fault. M is unchanged in the PTE mapping the first page. It is UNPREDICTABLE whether M is set in the PTE mapping the second page.

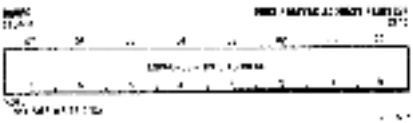
It is UNPREDICTABLE whether the modification of a process PTE(M) bit causes modification of the system PTE that maps that process page table. Note that the update of the M bit is not interlocked in a multiprocessor system.

<25> SBZ BIT
Bit 25 is reserved to DIGITAL and must be zero. The hardware does not necessarily test that this bit is zero because the PTE is established by privileged software.

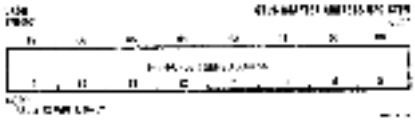
<24:23> OWNER BITS
Reserved for DIGITAL software use as the access mode of the owner of the page, (that is, the mode allowed to alter the page protection or to delete the page); not examined or altered by any hardware.

<22:21> SOFTWARE BITS
Bits 22 and 21 are reserved for DIGITAL software.

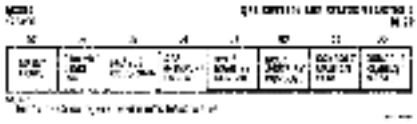
<20:00> PAGE FRAME NUMBER
The upper 21 bits of the physical address of the base of the page. Used by CPU hardware only if V=1.



Notes: See QADBI for register description.



The program loads the address in the address registers. It first loads the low-order byte in QADR <00> and then the high-order byte in QADRI. When the data is read it is stored in the data registers, QDATA0 and QDATA1 and CONSOLE ADDRESS PENDING is cleared. This signifies that the read operation is complete and that the data may be retrieved from QDATA0 and QDATA1.



<07>	MAINT MODE
	CL29 MAINT MODE H
	Set maintenance mode to allow simulation of QBus device.

<06>	QBA AND QBUS INIT
	CL29 QBA INIT H
	Initialize QBA and QBus devices.

<05>	ENABLE QBUS DMA
	CL29 ENA QBUS DMA H
	Enable QBus DMA transfer requests.

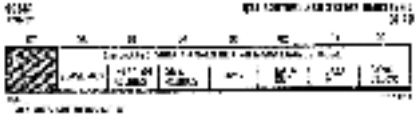
<04>	QBA INTERRUPT ENABLE
	CL29 QBA IE H
	Enable QBA to acknowledge QBus device interrupt request.

<03>	QBUS READ BY CONSOLE
	CL29 QBA READ H
	Request QBus read operation by Console.

<02>	QBUS WRITE BY CONSOLE
	CL29 QBA WRITE H
	Request QBus write operation by Console.

<01>	CONSOLE MASTER PEND
	CL31 CSL MASTER PEND H
	QBus read/write operation by Console in progress.

<00>	CONSOLE REQUEST MASTER
	CL31 CSL REQ MASTER H
	Request QBus for read/write operation by Console.



<07>	UNUSED

Bits <06:00> are used during maintenance only and simulate QBUS signals. If QCSR0 <07>, MAINT MODE, is set, the setting of QCSR1 <06:00> will answer the corresponding signal on the QBUS.

<06>	SIMULATE SLAVE ACK (QBUS SIGNAL:BSACKL)
CL29 SIM SACK H

<05>	SIMULATE INTERRUPT REQUEST (QBUS SIGNAL:BRIQ L)
CL29 SIM IRQ H

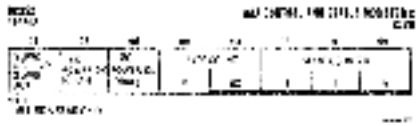
<04>	SIMULATE DMA REQUEST (QBUS SIGNAL:BDMR L)
CL29 SIM DMR H

<03>	SIMULATE REPLY (QBUS SIGNAL:BRPLY L)
CL29 SIM RPLY H

<02>	SIMULATE DATA OUT (QBUS SIGNAL:BDOUT L)
CL29 SIM DOUT H

<01>	SIMULATE DATA IN (QBUS SIGNAL:BDIN L)
CL29 SIM DIN H

<00>	SIMULATE SYNC CLOCK (QBUS SIGNAL:BSYNC L)
CL29 SIM SYNCH H



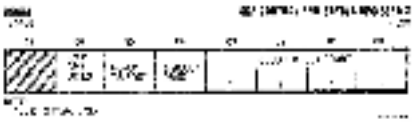
<07> SLAVE RECEIVED SLAVE ACK
CI:1X SRRC SACK 1
SACK received on QBus indicating QBus device is bus master.

<06> -AC POWER OK (QBUS)
-CL32 WRDK 0
When asserted, indicates the reception of QBus signal "-DRVOK H", which indicates a loss of AC power on the QBus.

<05> -DC POWER OK (QBUS)
-CL32 WRDK 0
When asserted, indicates the reception of QBus signal "-BRDOK H", which indicates a loss of DC power on the QBus.

<04:03> BYTE COUNT
CI33 <BitBU> 0
At the initiation of a QBus DMA sequence, this field counts the byte transfers required to translate a QBus word operation to a byte orientated data path to memory. Cleared by Console initialization.

<02:00> STATE COUNTER
CL49 <QAT 2:0> 0
Output of state generator in QBus control.



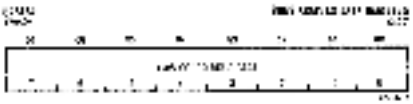
<07> NOT USED

<06> XMM CBUS DC LO L
      -CBUS DC LOW E
      This signal reflects the status of the DC LO signal from the XMM. When asserted (= 0), a DC LO condition exists.

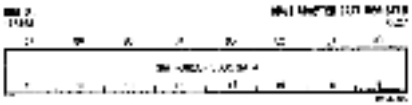
<05> RPLY TIMEOUT
      CL30 RPLY TIMEOUT H
      Reply not received on QBus during QBus read/write operation (H save). Interrupts Console (I-11) program.

<04> TIMEOUT
      CL3E TIMEOUT H
      QBus RPLY timeout counter in QBus control has timed out (H save).

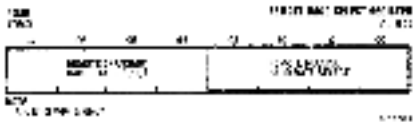
<03:00> QBUS TIMEOUT COUNT
      CL30 TIME <3:0> H
      Output of QBus RPLY timeout counter in QBus control.



Note: See QDATA1 for register description.



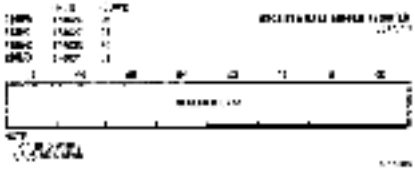
The data registers, QDATA0 and QDATA1, are loaded first. The console program then loads the address in the address registers. It first loads the low-order byte in QADR0 and then the high-order byte in QADR1. Loading the high-order byte in the address register causes the hardware to automatically begin the QBus operation at the PENDING MASTER STARTUP bit is set. The hardware automatically clears CONSOLE MASTER PENDING at the end of the QBus write. This indicates to the console program that the operation has ended.



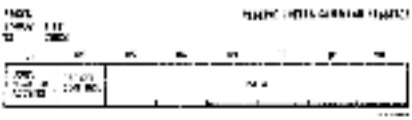
The T-11 program specifies the baud rate by loading the Console Baud Select Register (XBSR). The clock rates are set by loading two 4-bit codes (one for the transmit clock and the other for the receive clock).

XEMCYD BAUD SELECT REGISTER

BITS<7:4> BITS<3:0>	BAUD RATE	BITS<7:4> BITS<3:0>	BAUD RATE
0000	50	1000	1800
0001	75	1001	2000
0010	110	1010	2400
0011	134.5	1011	3600
0100	150	1100	4800
0101	300	1101	7200
0110	600	1110	9600
0111	1200	1111	19.2K



The XBUFC-3 registers are used in conjunction with the XBUFC register for transferring data packets from the console to the SBox ailtsands. This communication occurs only in Console I/O (CIO) mode and will take place between CSM and MCP (if in MACRO context) or BSM and DCP (if in DIAGNOSTIC context). Refer to CSM and DSM specifications for operation of these registers, as well as the "8600 Console Technical Description" manual, EK-KA86C-TD.

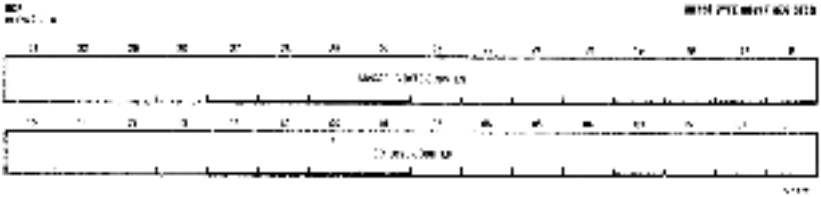


The XBUFC register is used in conjunction with the XBUFC-1 registers for communication between the console software and XBOX microcode. This communication occurs only in Console I/O (CIO) mode and will take place between CSM and XCP (if in MACRO context) or DSM and DCP (if in DIAGNOSTIC context). Refer to CSM and DSM specifications for operation of these registers, as well as the "SA86 Console Technical Description" manual, EK-SA86C-TD.

<07> DONE RIGHT OR ACCESS BIT
When DONE is set, the XBOX microcode may read the XBUF registers. When DONE is cleared, the I-11 program may write the XBUF register.

<06> PACKET CONTROL
When this bit is set, another packet will be sent by the XBox microcode. When it is not set, the current packet is the last one for this transaction.

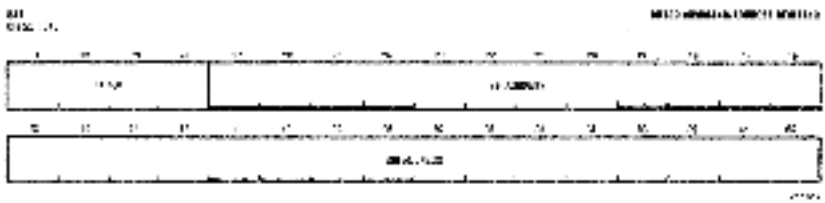
<05:00> DATA
Applications dependent; refer to DSM and CSM specifications.



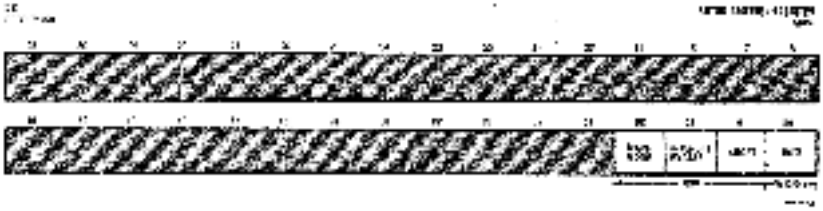
The program loads (writes) the 2's complement of the number of bytes to transfer into bits <15:00> of this register. Upon initiation of the transfer, the NBA hardware will load these 16 bits into <31:16> and into <15:00>. The Byte Count Register may not be modified during a data transfer. Any attempt to do so will be ignored and will result in a programming error, setting bit <13>, PGE, of the BI780 Status Register.

<31:16> MASSBUS BYTE COUNTER
The 2's complement byte count of the number of bytes to be transferred on the Massbus.

<15:00> SBI BYTE COUNTER
The 2's complement byte count of the number of bytes to be transferred on the SBI.



READ ONLY
This register is read only and valid only when CP BUSY is set. It contains the value of bits 31 through 00 of the SBI during the command/address part of the MBA's next data transfer.



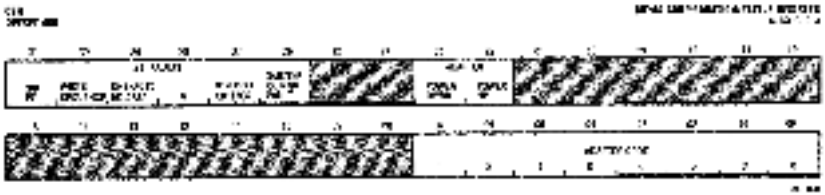
<31:04> RESERVED

<03> MAINTENANCE MODE
Set when the MBA is in the maintenance mode, which will allow the diagnostic programmer to examine and maintain the Massbus operations without a Massbus device. When set, the MBA will block RUN, GRM, and assert FAIL to the Massbus so that all the devices on the Massbus will detach from the Massbus. This bit can only be set if a data transfer is not in progress.

<02> INTERRUPT ENABLE
Set by writing a 1 or at power up. Cleared by writing a 0 or INIT. Setting this bit allows the MBA to interrupt the CPU when certain conditions occur.

<01> ABORT DATA TRANSFER
Set by writing a 1 and cleared by writing a 0, INIT or SSYNC. Setting this bit will initiate the data transfer abort sequence that will stop sending commands and addresses, and stop the byte counter. It will also negate RUN, assert EXC to the Massbus, wait for EBL, and set ABORT to a 1 at trailing edge of RBL. The CPU is interrupted if the interrupt Enable bit is 1.

<00> INITIALIZATION
This bit is self-clearing (always reads 0). Setting this bit clears status bits in the MBA configuration register, clears ABORT and IE bits in the MBA control register, clears the MBA status register, clears control and status bits of diagnostic registers. It will also cancel all pending commands except the read data pending short data transfer command as well as asserting the Massbus Init command.



NOTE

When bits 31, 30, 29, or 27 are set, FAULT will be asserted on the SBI for one cycle. In addition, they will be reset by a power fail or upon the deassertion of FAULT.

<31> SBI PARITY ERROR (FAULT A)
This bit is set when an SBI parity error is detected.

<30> WRITE DATA SEQUENCE (FAULT B)
This bit is set when a received command address indicates a write operation, but is not followed by write data.

<29> UNEXPECTED READ DATA (FAULT C)
This bit will be set when read data is received but is not expected (no read command address transmitted).

<28> RESERVED

<27> MULTIPLE TRANSMITTER FAULT (FAULT D)
When the ID received does not agree with the ID transmitted on the SBI, this bit will be set.

<26> TRANSMITTER DURING FAULT
This bit is set when the SBI fault is detected on the 2nd cycle after the MBA transmits information on the SBI. It is cleared by a power fail or the deassertion of FAULT on the SBI.

<25:24> RESERVED

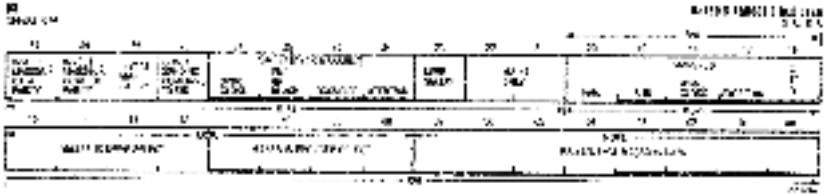
<23> ADAPTER POWER DOWN
This bit will be set when the MBA receives the assertion of AC LO. It will be reset when power comes up, by the assertion of INIT, UNIBA, DC LO, or the writing of a "1" to this bit. The setting of this bit will generate an interrupt if interrupt enable is set.

<22> ADAPTER POWER UP
This bit will be set when the MBA receives the deassertion of AC LO. It will be cleared upon power down, or by the assertion of INIT, UNIBA, DC LO, or the writing of a "1" to this bit. The setting of this bit will set the interrupt enable bit and interrupt the CPU.

<21> OVER TEMPERATURE
Always 0

<20:08> RESERVED

<07:00> ADAPTER CODE
The MBA adapter code is: <07:00> = 00100000



<31> INVERT MASSBUS DATA PARITY GENERATOR

<30> INVERT MASSBUS CONTROL PARITY GENERATOR

<29> INVERT MAP PARITY

<28> BLOCK SENDING COMMAND TO THE SDI
During a data transfer, the setting of this bit will eventually cause a DLT bit set and a DF ABORT.

<27> SIMULATE SCLK
If CR <31>, Maintenance Mode (MM) is set, this bit will simulate the assertion of SCLK.

<26> SIMULATE SBI
When the MM bit is set, this bit will simulate the assertion of SBI.

<25> SIMULATE DCC
When MM bit is set, this bit will simulate the assertion of DCC.

<24> SIMULATE ATTN
When MM bit is set, this bit will simulate assertion of ATTN.

<23> MAINTENANCE MASSBUS DATA INPUT BUFFER SELECT
When this bit is set, the upper eight bits (B<15:08>) of the WDIB will be sent all from bits 07 through 00 of the diagnostic register, if the diagnostic register is read. When the bit is cleared, the lower eight bits (B<07:00>) of the WDIB will be sent out from bits 07 through 00 of the diagnostic register, if the diagnostic register is read.

<22:21> MAINTENANCE USE ONLY
Read/write with no effect. Used to test the writability of these bits.

<20> MASSBUS FAIL (Read Only)
Fail is asserted when MM is set.

<19> MASSBUS RUN (Read Only)

<18> MASSBUS WCLK (Read Only)

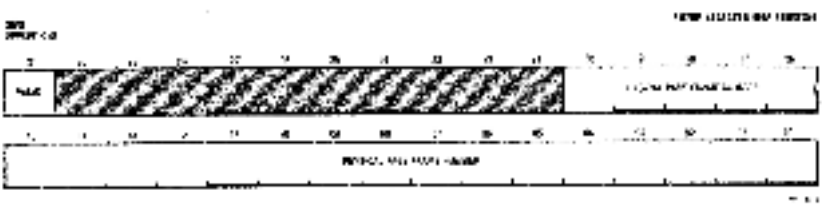
<17> MASSBUS EXC (Read Only)

<16> MASSBUS METOD (Read Only)

<15:13> MASSBUS DEVICE SELECT (Read Only)

<12:08> MASSBUS REGISTER SELECT (Read Only)

<07:00> MAINTENANCE UPPER/LOWER WDIB



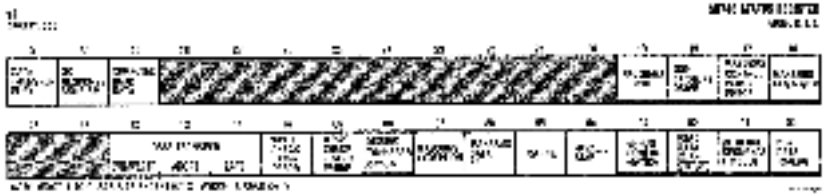
There are 256 MAP registers, and the Selected MAP Register (SMR) points to the contents of the MAP register pointed to by VAR <16:09>. The SMR is a read only register and is valid only when DT BUSY is set (GR <1>, data transfer command has been received).

The MAP registers can only be written when there is no data transfer operation in progress. A write to a MAP register while a data transfer is in progress will be ignored and cause the setting of PGE (SR <19>) and interrupt the CPU at the end of a transfer if IE is set.

<31> VALID BIT
Bits <20:00> is a valid physical page frame number.

<30:21> MBZ

<20:00> PHYSICAL PAGE FRAME NUMBER



NOTE

When set, the following bits will interrupt the CPU if IE is set: <19:17>, <10>, <3:1> and <00>. The following bits, when set, will cause the transfer to be aborted: <11:09> and <07:00>.

<31> DATA TRANSFER BUSY
This bit is set when a data transfer command is received. It is cleared when a data transfer is aborted. Read only.

<30> NO RESPONSE CONFIRMATION
This bit is set when the XBA received a no-response confirmation for a read or write command. It is cleared by writing a 1 to this bit or INIT. Setting of this bit will cause retry of the command.

<20> CORRECTED READ DATA (CRD)
Set when read data has a MASK field of 0001 (CRD), which indicates that the data has been corrected. It is cleared by writing a 1 to this bit or INIT.

<20:20> RESERVED

<19> PROGRAM ERROR
This bit is set when the program tries to: 1) Initiate a data transfer when a data transfer is already in progress; 2) Load MAP, VAR, or BC while a data transfer is in progress; 3) Set MBA maintenance mode during a data transfer operation; or Initiate an illegal data transfer command. It is cleared by writing a 1 to this bit.

<18> NON-EXISTING DRIVE
This bit is set when the drive fails to assert TRANSFER within 1.5 us after assertion of DEMAND. It is cleared by writing a 1 to this bit.

<17> MASSBUS CONTROL PARITY ERROR
This bit is set when a Massbus control parity error occurs. It is cleared by writing a 1 to this bit.

<16> ATTENTION FROM MASSBUS
This bit is set when the ATTENTION line on the Massbus is asserted.

<15:14> RESERVED

<13> DATA TRANSFER COMPLETE
This bit is set when data transfer is completed. It is cleared by writing a 1 to this bit.

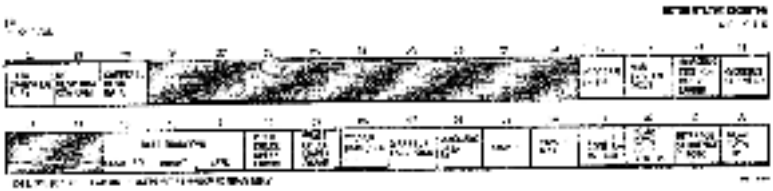
<12> DATA TRANSFER ABORTED
This bit is set with the trailing edge of MBE when the data transfer has been aborted. It is cleared by writing a 1 to this bit or INIT.

<11> DATA TRANSFER LATE
This bit is set 1) for either a write data transfer or a write check data transfer providing the data buffer is empty when WDCK is sent to the Massbus; 2) for a read data transfer providing the data buffer is full when RDCK is received from the Massbus.

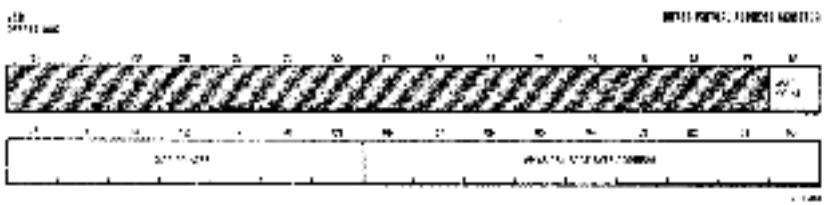
<10> WRITE CHECK UPPER ERROR
This bit is set when a compare error is detected in the upper byte while the MBA is performing a write check operation. It is cleared by writing a 1 to this bit or INIT.

<09> WRITE CHECK LOWER ERROR
This bit is set when a compare error is detected in the lower byte while the MBA is performing a write check operation. It is cleared by writing a 1 to this bit or INIT.

<08> MISSED TRANSFER ERROR
This bit is set when no SDC or NXM is received within 50 us after data transfer busy is set. It is cleared by writing a 1 to this bit or INIT.



<0:7>
<0:6>
<0:5>
<0:4>
<0:3>
<0:2>
<0:1>
<0:0>

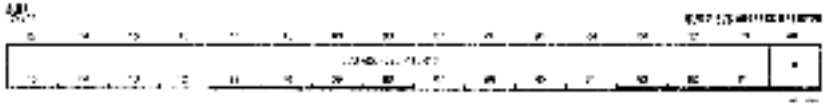


Notes: The Virtual Address Register should not be written to during a data transfer. An attempt to do so will set DSOCRAN EP800 (bit <15> in the KA780 Status Register). The virtual address register will not be modified and the data transfer will continue.

<31:17> RESERVED

<16:09> MAP POINTER
Select one of the 256 MAP registers.

<08:00> PHYSICAL PAGE BYTE ADDRESS
This field specifies the byte offset into the current (data) page. The contents of the selected MAP register and the value of this field are used to assemble a physical SBI address to be sent to memory.



The Bus Address Register is a 16 bit, word-addressable register with a standard address of 174402 for the 16 bit QBUS. Bits <1:00> can be read or written; bit <00> is usually written as 0. The Bus Address Register indicates the memory location for the DMA transfer during a read or write operation. The register's contents are automatically incremented by 2 as each word is transferred between the system memory and the controller.

Although unused on the VAX 8600 console, the Bus Address can be expanded for an 18 bit LSI-11 Bus using bits <05:04> of the BICSR (as UA <17:16>) or to a 22 bit LSI-11 Bus by using the BIBAR (BIIB12 Bus Address Extension) register bits <05:00>.

The BIBA register is cleared by initializing the Bus (BINIT L).



<15> COMPOSITE ERROR
When set, this bit indicates that one or more of the error bits (bits <14:10>) are set. Cleared by initializing the bus or starting a new command, with the exception of DE and ERR if they were caused by a drive error. Note: When an error occurs, the current operation terminates and an interrupt routine is started if the interrupt enable bit (bit 06 of the CSR) is set.

<14> DRIVE ERROR
When set, it indicates that the selected drive has flagged an error. The source can be determined by executing a get status command. Note: DE will not set ERR (bit 15) or CRDY (bit 07) until the usual occurrence of CRDY. This bit is buffered from the drive error interface line.

<13:10> CONTROLLER STATUS ERRORS
Set by the controller to indicate one of the following errors:

ERROR CODE BITS				ERROR	OCTAL CODE
13	12	11	10		
0	0	0	1	OPERATION INCOMPLETE (OPI)	1
0	0	1	0	DATA CRC (DCRC)	2
0	0	1	1	HEADER CRC (HCRC)	3
0	1	0	0	DATA LATE (DLT)	4
0	1	0	1	HEADER NOT FOUND (HNF)	5
1	0	0	0	NONEXISTENT MEMORY (NXM)	10
1	0	0	1	PARITY ERROR ABORT (PAR ERR)	11

Operation incomplete indicates that the current command was not completed within the OPI timeout period of 550 ms.

A data CRC error indicates that while reading the data field from the disk an error was found.

A header CRC error indicates that while reading the header from the disk an error was found. The CRC check is performed on the first and second header words, although the second header word is always 0.

Data late indicates that the FIFO DRM was more than half full and the controller was not able to read the next sequential sector. (This error may occur during a read without header check command.

Header not found indicates that an OPI timeout occurred while the controller was searching for the correct sector to read or write. A header compare did not occur.

A nonexistent memory error indicates that during a DMA transfer the memory location addressed did not respond with SACK within 10 us.

A memory parity error abort indicates that a parity error was detected while reading the system's optional memory that has parity error checking. If an error was detected, the current command to the RL11 is aborted.

<08:10> DRIVE SELECT
When set, these bits determine which drive will communicate with the controller via the drive bus. Cleared by INIT asserting the bus.

<07> CONTROLLER READY
Set by the controller at the completion of a command, at the detection of an error, or by initializing the bus. Cleared by software. This bit indicates that the command in bits 01-01 is to be executed. Note: Software cannot set this bit because registers are not accessible while CRDY is 0.

<06> INTERRUPT ENABLE
Set when CRDY is asserted, bit 06 allows the controller to interrupt the processor. Cleared by initializing the bus. Note: This interrupt occurs at the termination of a command. Once an interrupt request is placed on the LSI-11 bus, it is not removed until acknowledged by the LSI-11 processor even if it (bit 06) is cleared.

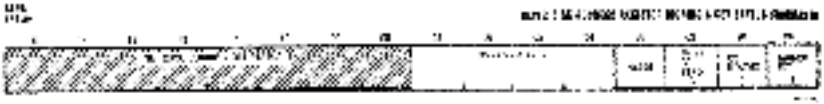
<05:04> EXTENDED ADDRESS BITS
These two bits are the upper-order bus address bits for 18-bit buses. These bits are read and written as bits 04 and 05 of the CSR. They function as address bits 16 and 17 of the BAR. Writing bits 04 and 05 of the CSR also writes bits 0 and 1 of the BAR.

<01:01> FUNCTION CODE
Set by software to indicate the command to be executed.

FUNCTION F3	F2	F1	COMMAND	OCTAL CODE
0	0	0	MAINTENANCE MODE	0
0	0	1	WRITE CHECK	1
0	1	0	GET STATUS	2
0	1	1	SEEK	3
1	0	0	READ HEADER	4
1	0	1	WRITE DATA	5
1	1	0	READ DATA	6
1	1	1	READ DATA WITHOUT HEADER CHECK	7

Cleared by initializing the bus (BINIT L). Value (command execution starts when CRDY (bit 07) of the CSR is cleared by software.

<00> DRIVE READY
When set this bit indicates that the selected drive is ready to receive a command or supply valid read data. Cleared when a seek or head select operation is started and set when the seek operation is complete.



The Disk Address Register is a 16 bit read/write register with a physical address of 174404. Its contents have one of three meanings, depending on the command being performed. The RLDA is cleared by initializing the bus (BINIT L).

RLDA during a GET STATUS command: Both the RLCSR and the RLDA registers must be programmed to execute the GET STATUS command.

<15:08> RESERVED

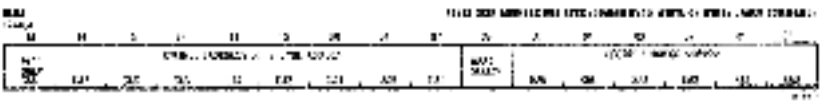
<07:04> MBZ
Must be zero.

<03> RESET
When this bit is set, the disk drive clears its error register of soft errors before sending a status word to the controller.

<02> MBZ
Must be zero.

<01> GET STATUS
Must be a 1, indicating to the drive to send its status word. At the completion of the get status command, the drive status word is read into the controller multipurpose register (RLMPR). With this bit set, bits 00-15 are ignored by the drive.

<00> MARKER
Must be a 1.



The Disk Address Register is a 16 bit Read/Write register with a standard address of 174404. The contents have one of three meanings, depending on the command being performed. The RLDA is cleared by initializing the bus (BINIT L).

RLDA during READ, WRITE or WRITE CHECK commands. For these commands, the RLDA register is loaded with head select information and the address of the first sector to be transferred. As each sector is transferred, the RLDA sector address increments by 1.

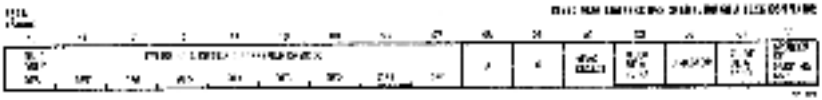
<15:07> CYLINDER ADDRESS
Address of one of the 256 cylinders for RL01 or 512 cylinders for RL02. (Octal range is 0 to 777.)

<06> HEAD SELECT
Indicates which read/disk surface is to be selected: 1 = lower, 0 = upper.

<05:00> SECTOR ADDRESS
Address of one of the 40 sectors on a track. (Octal range is 0 to 47.)

DADA (SEEK)

DADR (AFTER A GET STATUS)



The Disk Address Register is a 16 bit read/write register with a standard address of 17440A. Its contents have one of three meanings, depending on the command being performed. The DADA is cleared by initializing the bus (BINIT L).

DADA during a SEEK command: The program will execute a SEEK command by loading the DADA with the head select, direction to move and cylinder difference information.

<15:07> CYLINDER ADDRESS DIFFERENCE
Indicates the number of cylinders the heads are to move on a SEEK.

<06:05> RESERVED

<04> HEAD SELECT
Indicates which head (disk surface) is to be selected: 1 = lower, 0 = upper

<03> MBZ
Must be zero.

<02> DIRECTION
This bit indicates the direction in which the seek is to take place. The actual distance moved depends on the cylinder address difference (bits 07-15).

This bit is set when the heads move toward the spindle (to a higher cylinder address). It is cleared when the heads move away from the spindle (to a lower cylinder address).

<01> MBZ
Must be zero. Indicates to the drive that a SEEK command is being issued and that the other bits in the register have SEEK applications.

<00> MARKER
Must be a 1.



Following the GET STATUS command, a status word is stored in the DADR. The status word from the selected disk drive includes information on the functional state of the drive, including any drive errors.

<15> WRITE DATA ERROR
Indicates write data was asserted, but no pulses were detected on the write data line.

<14> HEAD CURRENT ERROR
Indicates that write current was detected in the heads when write gate was not asserted.

<13> WRITE LOCK
Indicates write lock status of selected drive: 0 = unlocked; 1 = protected.

<12> SEEK TIME OUT
Indicates that the heads did not arrive onto track within a specific time during a seek command.

<11> SPIN ERROR
Indicates that the spindle did not reach full speed within a specific time, or that it is turning too fast.

<10> WRITE GATE ERROR
Indicates that the write gate was asserted when the drive was not ready, the sector pulse was asserted, or the drive was write locked.

<09> VOLUME CHECK
VC is set every time the drive goes into load heads state. This asserts a drive error at the controller, but not on the front panel. VC is an indication that the program does not know which disk is present until it has read the serial number and bad sector file. (The disk might have been changed while the heads were unloaded.)

<08> DRIVE SELECT ERROR
Indicates multiple drive selection is detected.

<07> DRIVE TYPE
Indicates the type of disk drive: 0 = RL01, 1 = RL02.

<06> HEAD SELECT
Indicates the head selected: 1 = lower, 0 = upper.

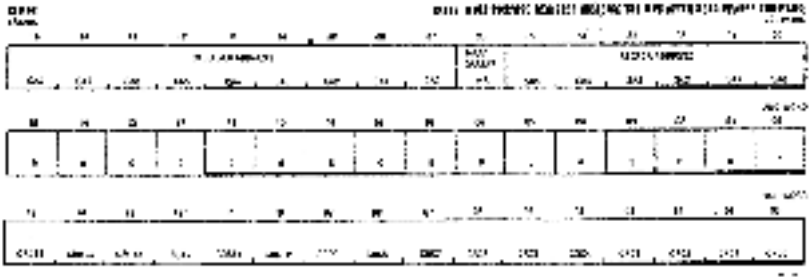
<05> COVER OPEN
Asserted when the cover is open or the dust cover is not in place.

<04> HEADS OUT
Asserted when the heads are over the disk.

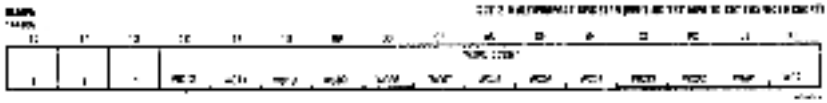
<03> BRUSH HOME
Asserted when the brushes are not over the disk.

<02:00> MAJOR STATE CODE BIT, BIT, BIT:

2	1	0	STATE OF DRIVE
0	0	0	LOAD STATE
0	0	1	SPIN UP
0	1	0	BRUSH CYCLE
0	1	1	LOAD HEADS
1	0	0	SEEK TRACK COUNTING
1	0	1	SEEK LINEAR MODE (LOCK ON)
1	1	0	UNLOAD HEADS
1	1	1	SPIN DOWN



After a READ HEADER command is executed, three words can be sequentially read from RLMPX. The first word includes the sector address, the head selected, and the cylinder address. The second word is all zeros. The third word is CRC computed over the header.



RLMPR loaded with a word count:

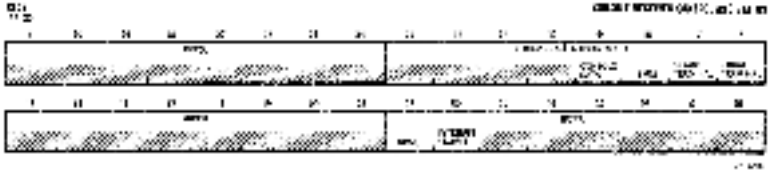
Before starting a DMA transfer, the RLMPR is loaded with the word count. The program must load the RLMPR with the 2's complement of the number of words to be transferred. The bits are described below. As each word is transferred, the RLMPR is automatically incremented by 1. The READ or WRITE operation continues until a word count overflow occurs, indicating that all words have been transferred.

The word count can range from 1 to 5120 data words. The maximum count is limited by the maximum number of sectors available (40) and the maximum words per sector (128).

Once written, the word count cannot be read back. Reading the RLMPR does not change the word count. The word count is cleared by initializing the Qbus (BINIT L).

<15:13> MUST BE ALL ONES
For word count in correct range.

<12:00> WORD COUNT
This is the 2's complement of the total number of words to be transferred. The maximum count is 5120.



<31:24> RXCS3 - RESERVED

<23:20> RXCS2 - DATA TERM READY
Reserved for Other Console Resident Interfaces.

<19:16> RXCS2 - DATA TERM READY
Logical Data Terminal Ready (DTR). Write only by CPU, initialized to zero by the console.

If any of the above bits (19-16) are set, the CPU is ready to accept data from the modem device. If any of the above bits (19-16) are not set, the CPU wants to disconnect from the modem.

<19> CONSOLE RBUA
Corresponds to "Logical" Console Subsystem Data.

<18> EMN
Corresponds to EMN Data Line.

<17> REMOTE TERMINAL
Corresponds to the remote services port on the console.

<16> LOCAL TERMINAL
Corresponds to console terminal.

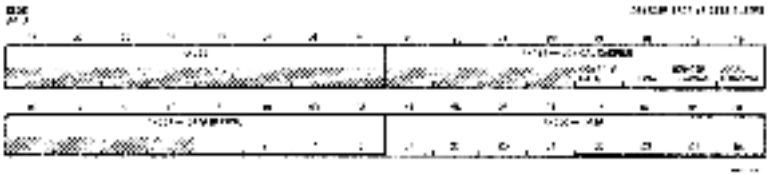
<15:08> RXCS1 - RESERVED

<07:00> RXCS0

<07> DONE
Read only by the CPU. Initialized to zero by the console. When set to a one, this bit indicates that data is available from the console, and that the data and the ID of the source of the data are both available in the RXDB.

<06> INTERRUPT ENABLE
Read/Write by the CPU. Initialized to zero by the console. When the logical AND of "RXCS DONE" and "RXCS IE" is equal to 1 a CPU interrupt is generated via System Control Block (SCB) vector "FC".

<05:00> RXCS0 - RESERVED



<31:24> RXDB3 - MBZ
Must be zero.

<23:20> RESERVED
Must be zero.

<19:16> RXDB2 CARRIER DESTINATION
Read only by the CPU. Initialized by the console based on the state of the corresponding DTR field <19:16> in RXCS and the state of any modem connection associated with that line.

<19> LOGICAL CONSOLE DATA
Because this is a hard-wired line (no modem involved) this bit will always be set.

<18> EMN
Because this is a hard-wired line (no modem involved) this bit will always be set.

<17> REMOTE TERMINAL (RTY)
Indicate that the modem associated with the Remote line (RTY) is active.

<16> LOCAL TERMINAL (CTY)
Because this is a hard wired line (no modem involved) this bit will always be set.

<15:12> RXDB1 - RESERVED
Must be zero.

<11:08> RXDB1 DATA IS CODE
Read only by the CPU. Initialized to zero by the console. Identifies the destination of the data byte in bits <07:00>.

ID	DESTINATION
00	Console Terminal
01	Remote Services Port
02	EMN Data
03	Logical Console Data
04	Reserved to DEC
.	.
.	.
0E	Reserved to DEC
0F	"Catcall" Byte Has Changed

<07:00> RXDB0 - DATA
Read only by the CPU. The Data Byte has different meanings depending of the destination specified by the DATA IN CODE <11:08>.

RXDB Data Byte Definitions:

RXDB DATA ID	Data Source
0	Console Terminal - RXDB <7:00> contains a byte of data received from the Console Terminal (CIT).
1	Remote Services Port - RXDB <7:00> contains a byte of data received from the Remote Services Port.
2	Environmental Monitoring Module (EMM) - In this case the RXDB data byte is used to identify the type of EMM data bytes that follow the RXDB.

The format for an EMM (RXDB) Data Byte is as follows:

7	6	5	4	3	2	1	0
TYPE	ASD	x	EMM OPCODE ID				

<07> Identifies the type of EMM Opcode contained in bits <04:00>. There are two types:

<07>	OPCODE TYPE
1	EMM EXCEPTION
0	EMM RESPONSE

EMM EXCEPTION OPCODES indicate that the EMM is reporting a status change. All Exception Reports consist of a single data byte which is also sent via RXDB.

EMM RESPONSE OPCODES indicate that the EMM is sending information in response to a request for information. Depending on the Response Opcode type, one or more bytes of additional information will be sent via RXDB.

<06> ASD (Automatic Shutdown) - Indicates that the EMM Automatic Shutdown Timer is running and that a total system power shutdown is pending. If the cause of the condition is not rectified (BAD SOME Temperature Fault or AIR FLOW Fault) the EMM will shutdown the system when the timer times out.

<05> RESERVED



RXDB DATA ID	Data Source
2	(Continued)
	<6:0,0> EXM OPCODE ID - Used in conjunction with bit <07> to identify the type of EXM data that follows in the next RXDB0 byte transfer. If set, <07> bit indicates that this field contains an EXM exception opcode and a single byte of data follows that describes the exception. Table 1 lists the EXM Exception Opcodes and describes the associated data byte. If clear <07> bit indicates that this field contains an EXM Response Opcode and one or more bytes of data follows that describes the exception. Table 2 lists the EXM Response Opcodes and describes the associated data bytes.
3	Logical Console Data. In this case RXDB0 contains a header ID that identifies the type of Logical Console data that follows via additional RXDB0 transmissions.

RDB DATA ID	BYTE	BITS	DESCRIPTION
10	00	<07:00>	Returned_Warm_Flag
	01	<08:01>	RESERVED
		<00>	0 = Warm_Flag Clear
			1 = Warm_Flag Set
11	00	<07:00>	Returned_Cold_Flag
	01	<08:01>	RESERVED
		<00>	0 = Cold_Flag = Warm
			1 = Cold_Flag Set
12	00	<07:00>	Returned UCODE Version
	01	<07:00>	Version Lower 8 Bits
	02	<07:00>	Version Upper 8 bits
13	00	<07:00>	Returned_Array_Config
	01	<07:00>	Array slots (7:1) in use
	02	<01:00>	Array type in slot 1
		<03:02>	Array type in slot 2
		<05:04>	Array type in slot 3
		<07:06>	Array type in slot 4

0X0E DATA ID	Data Source	HDR IB	DATA BYTE	BITS	DESCRIPTION
3	(Continued)				
			01	<01:00>	Array type in slot 5
				<03:02>	Array type in slot 6
				<05:04>	Array type in slot 7
				<07:06>	Array type in slot 8
					ARRAY CODES: ARRAY TYPE 0 Slot empty 1 16 Megabyte array 2 64 Megabyte array 3 Reserved
		20	0C	<07:00>	Console_Command_String_Complete The Console has successfully completed executing the CCS
		21	00	<07:03>	Returned_Last_VAX_PC
			01	<07:00>	VAX PC <07:00>
			02	<07:00>	VAX PC <15:08>
			03	<07:00>	VAX PC <23:16>
			04	<07:00>	VAX PC <31:24>
		22	0C	<07:00>	Returned_Last_CSM_Status
			01	<07:00>	CSM Status <07:00>
			02	<07:00>	CSM Status <15:08>
			03	<07:00>	CSM Status <23:16>
			04	<07:00>	CSM Status <31:24>
		30	00	<07:00>	SnapFile_Status_Returned The data byte indicates validity of the snapshot files
			01	<07:02>	RESERVED
				<00>	0 = SNAP1.DAT Invalid 1 = SNAP1.DAT Valid
				<01>	0 = SNAP2.DAT Invalid 1 = SNAP2.DAT Valid
		40	00	<07:00>	Console_Reboot_Successful Response to VME after a successful console reboot initiated by a MTPR CRBT.
		82	00	<07:00>	Console_Command_String_Error Indicates that: • the Console encountered an error while trying to execute an Console Command String (CCS) or, • the CCS was rejected because the CCS exceeded the 512 byte console buffer space or, • the console is locked via the terminal control switch

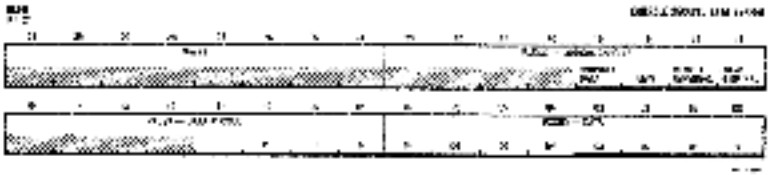


TABLE 1 - EMM EXCEPTION REPORTS

Use this table to interpret the meaning of the Opcode in bits <04:00> if bit <07> is set. In addition to listing the Opcodes, this table also describes the meaning of the contents of the data byte associated with each Opcode.

OPCODE	EXCEPTION TYPE
00	Status Change in Regulator A +5v The contents of the Data Byte associated with this and Opcodes 0 through A indicates that the status has changed in the following manner:

DATA BYTE	BITS	MEANING
1	<07:01>	RESERVED
	<00>	0 = Voltage Normal 1 = Voltage out of Spec.

01	Status Change in Regulator B +5v Data Byte definition: See OPCODE 00.
02	Status Change in Regulator C +5v Data Byte definition: See OPCODE 00.
03	Status Change in Regulator D -2v Data Byte definition: See OPCODE 00.
04	Status Change in Regulator E -2v Data Byte definition: See OPCODE 00.
05	Status Change in Regulator F -5.2v Data Byte definition: See OPCODE 00.
06	Status Change in Regulator H -5.2v Data Byte definition: See OPCODE 00.
07	Status Change in Regulator L +12v Data Byte definition: See OPCODE 00.
08	Status Change in Regulator L -12v Data Byte definition: See OPCODE 00.
09	Status Change in Regulator M +15v Data Byte definition: See OPCODE 00.
0A	Status Change in Regulator N -15v Data Byte definition: See OPCODE 00.

TABLE 1 - KMN EXCEPTION OPCODES (continued)

OPCODE#	EXCEPTION CAUSE
0A	Status Change in T1 Temperature The Data Byte associated with the OPCODE ID's 8 through A indicates the following status change: DATA BYTE / BITS / MEANING 7 / <07:02> / RESERVED <01:00> / 0 = Temperature Normal 1 = Temperature in Yellow Zone 2 = Temperature in Red Zone. This will result in an automatic shutdown if not corrected. 3 = Temperature Below Normal Range
0C	Status Change in T2 Temperature Data Byte Meaning: See ID 0A.
0D	Status Change in T3 Temperature Data Byte Meaning: See ID 0A.
0E	Status Change in T4 Temperature Data Byte Meaning: See ID 0A.
0F	Status or Delta T2-T1 has Changed The Data Byte associated with the OPCODE ID's E through 11 indicates the following status change: DATA BYTE / BITS / MEANING 7 / <07:02> / RESERVED <01:00> / 0 = Difference Normal 1 = Difference in Yellow zone 2 = Difference in Red Zone. This will result in an automatic shutdown if not corrected.
10	Status of Delta T3-T1 has Changed Data Byte Meaning: See ID 0F.
11	Status of Delta T4-T1 has Changed Data Byte Meaning: See ID 0F.
12	Status of AIR FLOW SENSOR 1 has Changed The Data Byte associated with the OPCODE ID's 12 and 13 indicates the following status change: DATA BYTE / BITS / MEANING 1 / <07:01> / RESERVED <00> / 0 = Air Flow Normal 1 = Air Flow out of spec. This will result in an automatic shutdown if not corrected.
13	Status of AIR FLOW SENSOR 2 has Changed Data Byte Meaning: See ID 12.

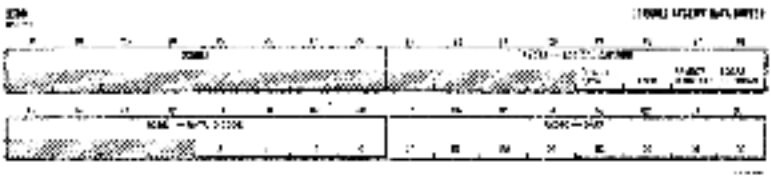


TABLE 1 - EBM EXCEPTION REPORTS (continued)

OPCODE	EXCEPTION TYPE

14 Status of CPU Availability has Changed. Byte 0 associated with this ID has the following meaning:

DATA BYTE	BITS	MEANING
1	<07:02>	RESERVED
	<00>	0 = CPU available 1 = CPU not available

15 EBM STATUS CHANGED - Except for the case where the EBM is dead, the EBM is rebooted by the console when any of the following errors occur.

1	<07:04>	RESERVED
	<03:00>	0 = EBM Dead (failed to restart) 1 = EBM encountered XBM parity error 2 = EBM encountered an illegal instruction 3 = EBM encountered an unknown trap to C 4 = EBM encountered an unexpected trap 5 = EBM encountered an unexpected 6.5 interrupt 6 = Excessive collisions on EBM bus 7 = No transport acknowledge from EBM 8 = No response from EBM 9 = Negative response from EBM A = EBM has no buffers available B = CSL to EBM message transmit timeout

16 TX RDY TIMEOUT - The TXCS RDY bit has not been set by the console for 2 seconds. Operation in progress aborted.

DATA BYTE	BITS	MEANING
1	<07:01>	RESERVED
	<01:00>	0 = Local terminal operation aborted 1 = Remote services port operation aborted 2 = EBM operation aborted 3 = Logical console operation aborted

TABLE 2 - EMM RESPONSE SYNTAX

Use this table to interpret the meaning of the Opcode in bits <04:00> if bit <07> is clear. In addition to listing the Opcodes, this table also describes the meaning of the contents of the data bytes associated with each Opcode.

NOTE

The first data byte following the RXDB is always a byte count.

IN	INDICATION
00	RESPONSE TO 'EMM STATUS' - This operation returns the status of the EMM status registers and PROM revision number. This response consists of the following 8 Data Bytes.

BYTE	BITS	MEANING
0	<07:00>	Packet size = 8 bytes (10)
1		POWER CONTROL REGISTER
	<00>	Regulator A +5V BSU *
	<01>	Regulator C +5V SBIA *
	<02>	Regulator B -2V ECL *
	<03>	Regulator E -2V ECL *
	<04>	Regulator F +5.2 ECL *
	<05>	Regulator H +5.2 ECL *
	<06>	Regulator J (unused) * 0 = regulator off
	<07>	SBU disable ** ** 0 = enabled
2		MARGIN ENABLE REGISTER
	<00>	Regulator A +5V CSL *
	<01>	Regulator D +5V SBU *
	<02>	Regulator C +5V SBIA *
	<03>	Regulator B -2V ECL *
	<04>	Regulator E -2V ECL *
	<05>	Regulator F +5.2 ECL *
	<06>	Regulator H +5.2 ECL *
	<07>	Regulator J (unused) * 1 = Margin Enabled
3		MARGIN SELECT REGISTER
	<00>	Regulator A +5V CSL *
	<01>	Regulator D +5V SBU *
	<02>	Regulator C +5V SBIA *
	<03>	Regulator B -2V ECL *
	<04>	Regulator E -2V ECL *
	<05>	Regulator F +5.2 ECL *
	<06>	Regulator H +5.2 ECL *
	<07>	Regulator J (unused) * 1 = Margin Enabled
4		MONOX REGISTER
	<00>	Regulator A -5V CSL *
	<01>	Regulator E +5V SBU *
	<02>	Regulator C +5V SBIA *
	<03>	Regulator D -5V ECL *
	<04>	Regulator E -2V ECL *
	<05>	Regulator F +5.2 ECL *
	<06>	Regulator H +5.2 ECL *
	<07>	Regulator J (unused) * 1 = MOD OK

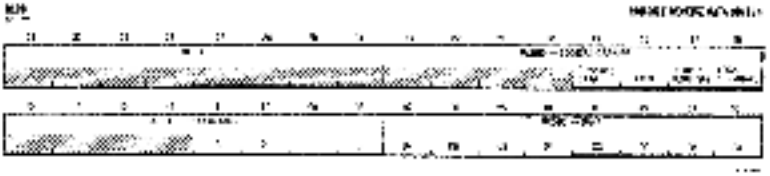


TABLE 2 - MMB RESPONSE REPORTS (continued)

ID	INDICATION
80	(Continued)

BYTE	BIT	MEANING
5		MMB80 REGISTER
	<00>	Regulator K OK *
	<01>	Regulator L OK *
	<02>	Module K AC LO
	<03>	Module L AC LO
	<04:06>	MMB unit number **
	<07>	Key OVERRIDE circuit status * 1 = MOD OK ** 0 = KA86xx
6		MISCELLANEOUS HARDWARE STATUS REGISTER
	<00>	Air Flow 1 Sensor Status (1 = fault)
	<01>	BBU unit (1 = failure)
	<02>	Minus 2V CROWBAR fault (1 = crowbar)
	<03>	Air Flow 2 Sensor Status (1 = fault)
	<04>	Latched AC LO (1 = AC LO)
	<05>	Latched DC LO (1 = DC LO)
	<06>	Parity Bit
	<07>	Parity Error
7		MISCELLANEOUS SOFTWARE STATUS REGISTER
	<00>	Reserved Output Signal (1 = asserted)
	<01>	Default Node Enable (1 = enabled)
	<02>	Program Abort (1 = active)
	<03>	Interrupt Disable (1 = disabled)
	<04:07>	Reserved
8		MMB PROM VERSION NUMBER
	<07:00>	Integer value of the MMB PROM version

8* Response to System Environment Information Request. This operation returns 32 bytes which reflect all measured environmental information, which includes the voltage outputs of all regulators and temperature measurements at all sensors.

TABLE 7 - XMI RESPONSE REPORTS (continued)

ID	INDICATION
91	(Continued)

NOTE

Values returned are in their digitized format and must be converted using the formulas below before being displayed or interpreted. The number in parentheses for each entry indicates which equation must be used to perform the conversion.

1. Volts = .0276 * MAGNITUDE

2. Milliamps = 70 - (2.9 * MAGNITUDE)

3. Volts = .0962 * MAGNITUDE

4. Volts = .069! * MAGNITUDE

5. Degrees Celsius = 16 + (.3333 * MAGNITUDE)

* All numbers in equations are decimal.

BYTE	BITS	MEANING
0	<07:00>	Packet size = 32 bytes (10)
1	<07:00>	REGULATOR A (+5V TTL) Magnitude See note 1.
2	<7>	REGULATOR A (+5V TTL) Polarity 0 = Positive Polarity 1 = Negative Polarity
	<06:00>	Reserved
3	<07:00>	REGULATOR B (+5V ECL) Magnitude See Note 1.
4	<7>	REGULATOR B (+5V ECL) Polarity See Byte 2
	<06:00>	Reserved
5	<07:00>	REGULATOR C (+5V SBIA) Magnitude See Note 1.
6	<7>	REGULATOR C (+5V SBIA) Polarity See Byte 2
	<06:00>	Reserved
7	<07:00>	REGULATOR D (-2V ECL) Magnitude See Note 1.
8	<7>	REGULATOR D (-2V ECL) Polarity See Byte 2
	<06:00>	Reserved
9	<07:00>	REGULATOR E (-2V ECL) Magnitude See Note 1.

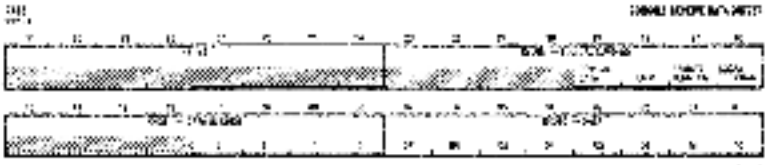


TABLE 2 - EMM RESPONSE REPORTS (continued)

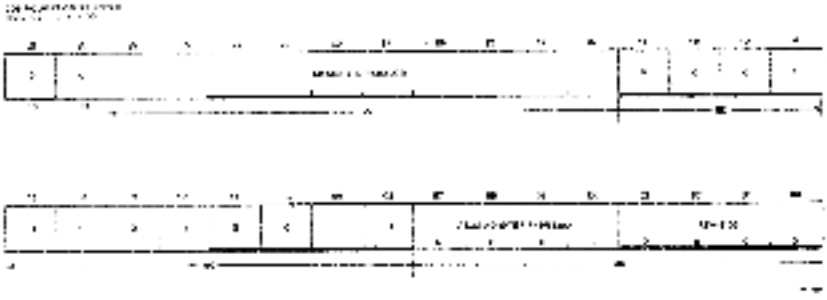
ID	INDICATION
01	(Continued)

BYTE	BITS	MEANING
10	<07>	REGULATOR E (-2V ECL) Polarity See Byte 2
	<06:00>	Reserved
11	<07:00>	REGULATOR F (-5.2V ECL) Magnitude See Note 1.
12	<07>	REGULATOR F (-5.2V ECL) Polarity See Byte 2
	<06:00>	Reserved
13	<07:00>	REGULATOR G (-5.2V ECL) Magnitude See Note 1.
14	<07>	REGULATOR H (-5.2V ECL) Polarity See Byte 2
	<06:00>	Reserved
15	<07:00>	GROUND CURRENT Magnitude See Note 2.
16	<07>	GROUND CURRENT Polarity 0 = Positive Polarity 1 = Negative Polarity
	<06:00>	Reserved
17	<07:00>	REGULATOR L (+12V TTL) Magnitude See Note 3.
18	<07>	REGULATOR L (+12V TTL) Polarity Always Positive (i.e., 0)
	<06:00>	Reserved
19	<07:00>	REGULATOR L (-12V TTL) Magnitude See Note 4.
20	<07>	REGULATOR L (-12V TTL) Polarity Always Negative (i.e., 1)
	<06:00>	Reserved
21	<07:00>	REGULATOR K (+15V TTL) Magnitude See Note 3.
22	<07>	REGULATOR K (+15V TTL) Polarity Always Positive (i.e., 0)
	<06:00>	Reserved

TABLE 2 - ENB RESPONSE RESOURCE (continued)

ID	NOTICATION
01	(Continued)

BYTE	BITS	MEANING
23	<07:00>	REGULATOR A (-15V ITL) Magnitude See Note 5.
24	<07>	REGULATOR A (-15V ITL) Polarity Always Positive (i.e., 1)
	<06:00>	Reserved
25	<07:06>	TERMINATOR (T1) Magnitude See Note 3.
26	<07>	TERMINATOR (T1) Polarity Always Negative (i.e., 1)
	<06:00>	Reserved
27	<07:00>	TERMINATOR (T2) Magnitude See Note 3.
28	<07>	TERMINATOR (T2) Polarity Always Negative (i.e., 1)
	<06:00>	Reserved
29	<07:06>	TERMINATOR (T3) Magnitude See Note 3.
30	<07>	TERMINATOR (T3) Polarity Always Negative (i.e., 1)
	<06:00>	Reserved
31	<07:00>	TERMINATOR (T4) Magnitude See Note 3.
32	<07>	TERMINATOR (T4) Polarity Always Negative (i.e., 1)
	<06:00>	Reserved



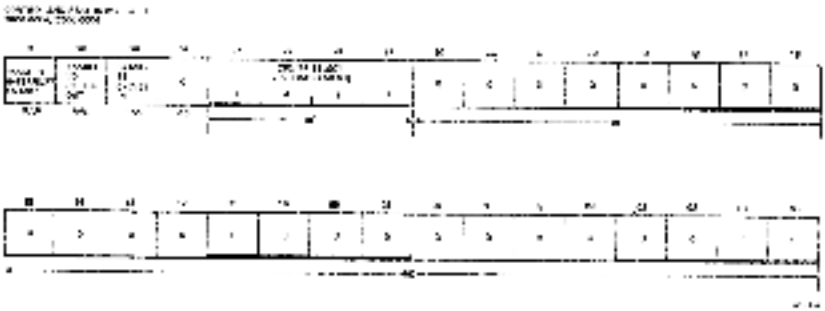
<31:30> MBZ (8820)
Must be zero.

<29:20> MEMORY SEPARATOR
8820 MSR <17:16>
This field defines the memory address boundry and is equal to the number of megabytes of memory addressable over the ABUS. If bit 29 is asserted, there are 512 MBytes of memory and bits <28:20> are disregarded when the hardware checks the DMA address. These bits are bits <29:20> in the Memory Separator register, but within the SBIA they are shifted right by two bits to match the physical address.

<19:08> MBZ (8530)
Must be zero.

<07:04> ABUS ADAPTER TYPE
8CS REG D<07:04> (8528)
These bits identify the type of ABUS adapter. 0001 for the SBIA.

<03:00> ABUS ADAPTER REVISION
SBI REG D<03:00> (8528)
These bits identify the revision of the ABUS adapter. These bits are hardwired.



<31> MASTER INTERRUPT ENABLE
SS29 MSTR INTR ENA
When set, this bit will enable the SBIA module to prioritize the interrupts and generate the appropriate interrupt priority level for CPU polling.

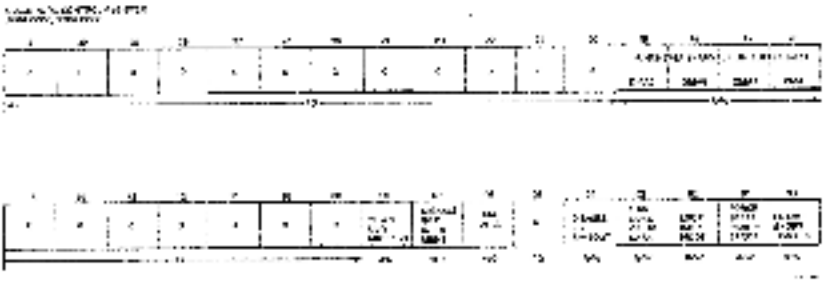
<30> SBIA SBI CYCLES OUT
SS29 SBIA SBI OUT
This bit must be set for normal operation. It enables the CPU to access SBI SBIA5 registers. If the CPU attempts to access an SBI SBIA5 register with this bit reset, it is an illegal condition, and ERROR SUMMARY register <20> and <19> will be set. See description of ERROR SUMMARY register.

<29> SBIA SBI CYCLES IN
SS29 SBIA SBI IN
This bit must also be set for normal operation. It enables all SBIA activity through the SBIA. If this bit is not set, the SBIA will not recognize SBI function codes and will not respond to SBI commands (SBI confirmation is OD, no response).

<28> ZERO (SS33)
This read only bit will always be zero.

<27:24> CPU TR SELECT <03:00>
CPU TR SEL <03:00> (SS07)
These bits provide backplane visibility of the register used to select the SUI TR for CPU transactions. The contents of this field is the two's complement of the TR level.

<23:00> ZERO ($S06)
These bits are always read as zero provided by the zero bit logic.



<31:30> MBZ (8536)
Must be zero.

<29:16> FORCE DMA TRANSACTION BUFFER BUSY
8529 FORCE DMAC (DMAU, DMAA, DMAI) BUSY
These bits are used to direct DMA traffic into specific DMA transaction buffers by forcing other buffers to be busy. The state of these bits has no effect on a DMA transaction already in progress.

<15:09> MBZ (8542)
Must be zero.

<08> CLEAR SILO ADDRESS
8519 CLR SILO ADR
This bit will clear the silo address upon setting. When this register is read, this bit will always be zero (hardwired).

<07> DISABLE SILO INCREMENT
8529 DISABLE SILO INC
When set, this bit will prevent the silo address from incrementing. This bit will be reset during normal operations to allow the silo address to increment. This bit is also read as zero.

<06> DIAGNOSTIC DEAD
8529 DIAG DEAD
When this bit sets, it will simulate ABUS DEAD, interrupting the console and causing a reboot. ABUS DEAD is normally asserted by SBI FAIL. This bit is also read as zero.

<05> MBZ (8530)
Must be zero.

<04> DISABLE SBI TIMEOUT
8529 DISABLE SBI TMO
When set for diagnostics, this bit will prevent a timeout condition while waiting for the SBIA to gain control of the SBI, while waiting an acknowledgment from a NAXAM, or while waiting for CPU read data.

<03> FORCE QUADWORD DATA
8529 FORCE QUAD DATA
This bit is used by microdiagnostics with bit <0> (Force Back Wonk), to provide a way to use a quadclear to loop data back on the SBI. FORCE QUAD DATA is set, then the CPU will execute a quadclear, but for microdiagnostics, the address will be a memory address instead of an SBI address (bit 27 is clear).

The ABus command/address is the same; it specifies a CPU write to the quadclear register. The ABus write data is the same except for the address, which will be for a memory (cache) address. When the command/address is transmitted on the SBI it will be received by the SBIA, as it always is, but in this case, the address will be less than the configuration register. To the SBIA it looks like a DMA extended write mask to memory and it will be handled as such.

For a normal quadclear, the write data is forced to all zeros. In this case, FORCE QUAD DATA will enable the A-Data Assembly Multiplexers to transfer the contents of the Write Data Latch (the ABus Write Data, 1011 and the Quadword Boundary Address) to the SBI. When the second write data longword is transferred to the ABI transceivers, bits 30 and 27 will be toggled (not). This will allow the testing of all data bits on the SBI.

<02> LOOP BACK MODE
SS29 LOOP BACK MODE
This bit is used by microdiagnostics to allow a CPU read or write to be looped back in the SBIA. The PAMM has to be configured such that a memory (cache) address is mapped to an I/O adapter, and the same PAMM address, but with bits 27 and 28 inverted, is mapped to a memory address.

In the SBIA, LOOP BACK MODE will invert address bits 25 and 26, if bit 27 is reset, which will be the case if the CPU write is to a memory address.

When the CPU writes a memory location that is mapped to an I/O adapter, the MBox will write the command/address and write data longword into the register file. The SBIA will carry out the command like a normal CPU write. When the command/address is transferred from the command/address latch to the SBI, because LOOP BACK MODE is set and address bit 27 is reset, address bits 25 and 26 will be inverted.

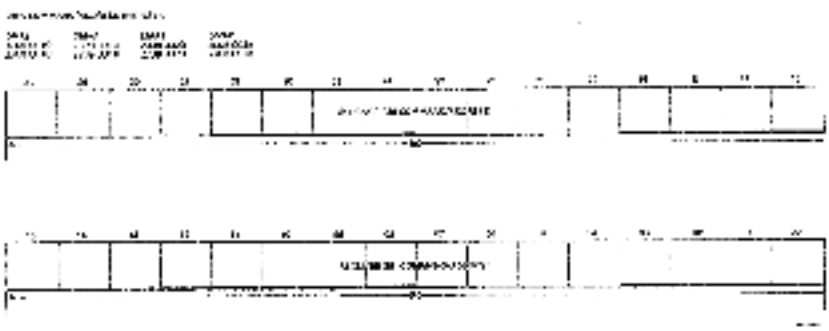
The command/address, followed by the write data will be transmitted on the SBI. When the SBIA clocks the SBI receivers and looks at the received data, the address is less than the memory separator (in the configuration register), it will transfer the command/address and write data to the register file and request MBox service.

The MBox will write the data into memory because the address, with bits 27 and 28 inverted (bits 25 and 26 in the SBIA), addresses a different PAMM location. This location is mapped to memory.

This diagnostic bit can also be used with a CPU read in a similar manner to further check out the SBIA logic.

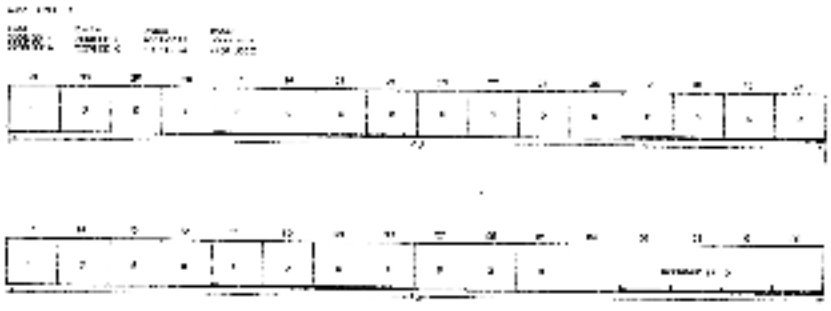
<01> FORCE STATE PARITY ERROR
SS29 FORCE STATE PTY
If this bit is set, a state machine parity error will be forced during the CPU ARB WAIT state.

<00> ENABLE SHORT TIMEOUT
SS29 ENA SHORT TIMEOUT
When set, this bit will enable an SBI timeout on 2 SBI cycles instead of the normal 512 cycles. This bit is read as a zero.



These four registers (DMAACA, DMABCA, DMACCA, and DMAICA) are used to save information about transactions in the DMA buffers. Each time a command/address is loaded into a DMA buffer, a copy of the command/address and SBI ID of the commander is saved in the two registers corresponding to the DMA buffer. If an error is detected by the MBox or the SBIA after the transaction is continued, the DMA registers are locked.

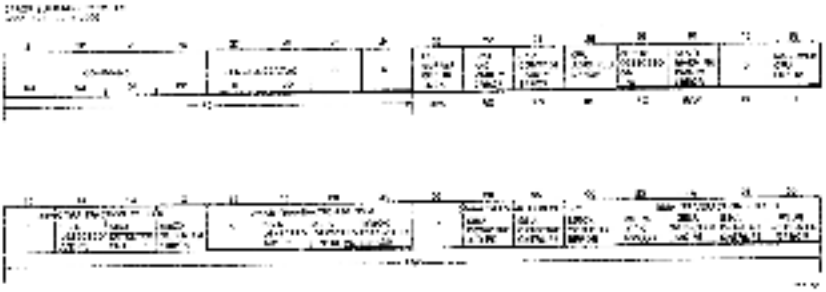
<31:00> RECEIVED SBI COMMAND/ADDRESS
SDS REG <31:00> (8532, 8833)
Each time a command/address is loaded into a DMA transaction buffer in the DC022, that command/address is also loaded into the corresponding DMA Command/Address error register. These error registers are actually ITL register files that are addressed by the upper two bits of the DC022 write address. SBI B <31:00> are written in these registers, with bits <31:28> being the SBI command codes and bits <27:00> the longword address. These error registers are locked if the SBIA or MBox detects a DMA error.



These four registers (DMAAIA, DMAAIB, DMADIA, and DMADIB) are used to save information about transactions in the DMA buffers. Each time a command/address is loaded into a DMA buffer, a copy of the command/address and SBI id of the commander is saved in the two registers corresponding to the DMA buffer. If an error is detected by the MBox or the SBIA after the transaction is confirmed, the two registers are locked.

<31:08> RSV (RSV36)
Must be zero.

<07:00> RECEIVED SBI IDENTIFICATION
BUS REQ <07:00> (RS23)
Each time a command/address is loaded into a DMA transaction buffer in the DC022, the SBI id is also loaded into the corresponding DMA ID error register, an extension of the DMA command/address error registers. These error registers are TTL register files like the command/address error registers, and addressed the same way, by the upper two bits of the DC022 write address. Bits <07:02> have the inputs at ground potential so they will always be read as zero. Bits <04:00> are loaded with DEC SBI ID <04:00>. Like the DMA command/address error registers, these registers are locked if the SBIA or MBox detects a DMA error.



The error summary register contains most of the information about errors which have been detected by the SBIA on transactions involving the SBIA.

<31:28> COMMAND <03:00>
BUS REG D<31:28> (SS26)
These bits represent the ABus command bits for a CPU I/O register read/write. They are loaded each time the command/address latch is loaded, and are latched by CPU ERROR LOCK, bit <23>.

<27:26> LENGTH/STATUS <01:00>
BUS BRG <27:26> (SS26)
These bits represent the ABus data length for a CPU I/O register read/write. They are also loaded each time the command/address latch is loaded, and are latched by CPU ERROR LOCK, bit <23>.

<25:24> MBZ
Must be zero.

<23> CPU BUFFER ERROR LOCK
SSYY CPU ERROR LOCK
This bit will be asserted for any of the following errors on a CPU I/O read/write.

1. A/D Parity Error (bit <22>)
2. Control Parity Error (bit <21>)
3. Address Error (bit <20>)
4. CPU read/write timeout on SBI (SBI Error Register bit <12>)
5. SBI Error (SBI Error Register <08>)

If this bit is set, Error Summary Register <31:26> and the Timeout Address Register are latched. If clear, these bits will represent the most recent transaction. Writing this bit will clear Error Summary Register <22:19,16>.

<22> CPU A/D PARITY ERROR
SBAN A/D PTY BAD
This bit is set if a parity error is detected on the address/data bits of the command/address or write data for a CPU I/O read/write. If the error is detected on the command/address, bit <19> will also be set. Parity is checked on the output of the file data latch. If this bit sets, bit <23> is set. This bit is cleared when the CPU writes bit <23>.

<22> CPU CONTROL PARITY ERROR
SBAR CNTRL PTY BAD
This bit is set if a parity error is detected on the control field of the command/address or write data for a CPU I/O read/write. If the error is detected on the command/address cycle, bit <19> will also be set. Parity is checked on the output of the fill data latch. If this bit sets, bit <23> is also set. This bit is also cleared when the CPU writes bit <23>.

<20> CPU ADDRESS ERROR
SBIE LOCAL ADR ERR
This bit is set if the CPU accesses a nonexistent SBIA register or when an SBI NXM06 register is accessed when Control/ Status Register <30> is clear (CPU access to the SBI is disabled). When it sets, it will set bit <23>, and it is cleared when the CPU writes bit <23>. This error will be detected when the command/address word is available, so bit <19> should also be set.

<19> ERROR DETECTED ON COMMAND/ADDRESS CYCLE
SBAR ERR ON C/A
This read only bit is set if a address/data, control parity, or address error is detected on the command/address cycle. The setting of this bit will set bit <23>. This bit will be reset when the CPU writes a one to bit <23>.

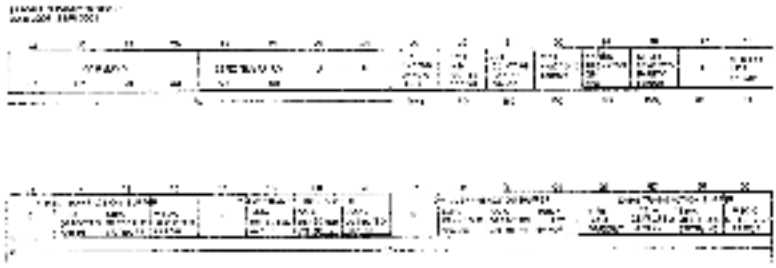
<18> STATE MACHINE PARITY ERROR
SBAO FORCE PARITY ERAP
This error bit will be set if the state machine microword does not contain even parity. The occurrence of this error will cause a CPU transaction to be aborted, if one is in progress, and generate an interrupt. A state machine parity error can occur if no CPU transaction is in progress so it will not set bit <23>.

<17> MBZ (SS33)
Must be zero.

<16> MULTIPLE CPU ERROR
SBAR MULT CPU ERR
This bit can only be set if bit <21> is already set and a CPU addressing error is detected on the command/address cycle or there is an address/data or control parity error on the command/address or write data. This bit will not be set for a write data parity error for the transaction that sets bit <23>, but for a subsequent transaction. Bit <16> is reset when the CPU writes bit <23>.

<15> MBZ (SS32)
Must be zero.

<14> DMAC TRANSACTION BUFFER SBIA DETECTED A/D PARITY ERROR
SBIU DMAC A/D ERROR
This bit will be set for a data parity error when the read data is being transferred from transaction buffer 0 to the SBI during a DMA read. This bit cannot be set if bits <13> or <12> have been previously set. This bit is cleared by the CPU writing it. The DMAC Command/Address Register and DMAC ID Register will be locked if this bit sets. The setting of this bit will generate a local interrupt.



<13> DMAC TRANSACTION BUFFER SBIA DETECTED CONTROL PARITY ERROR
SBIB DMAC CONTROL ERROR
This bit will be set for a control parity error when the read data is being transferred from transaction buffer C to the SBI during a DMA read. This bit cannot be set if bits <14> or <12> have been previously set. This bit is cleared by the CPU writing it. The DMAC Command/Address Register and DMAC ID Register will be locked if this bit sets. The setting of this bit will generate a local interrupt.

<12> DMAC TRANSACTION BUFFER MBOX DETECTED ERROR
SBIB DMAC MBOX ERR
This bit will be set if the MBox detects a parity error or HERR on the transfer of command/address from the DMAC transaction buffer. This bit cannot be set if bits <14> or <13> have been previously set. This bit is cleared by the CPU writing it. The DMAC Command/Address Register and DMAC ID Register will be locked if this bit sets. The setting of this bit will generate a local interrupt.

<11> MBZ
Must be zero.

<10> DMAB TRANSACTION BUFFER SBIA DETECTED A/D PARITY ERROR
SBIB DMAB A/D ERROR
This bit will be set for a data parity error when the read data is being transferred from transaction buffer B to the SBI during a DMA read. This bit cannot be set if bits <09> or <08> have been previously set. This bit is cleared by the CPU writing it. The DMAB Command/Address Register and DMAB ID Register will be locked if this bit sets. The setting of this bit will generate a local interrupt.

<09> DMAB TRANSACTION BUFFER SBIA DETECTED CONTROL PARITY ERROR
SBIB DMAB CONTROL ERROR
This bit is set for a control parity error when the read data is being transferred from transaction buffer B to the SBI during a DMA read. This bit cannot be set if bits <10> or <08> have been previously set. This bit is cleared by the CPU writing it. The DMAB Command/Address Register and DMAB ID Register will be locked if this bit sets. The setting of this bit will generate a local interrupt.

<08> DMAB TRANSACTION BUFFER MBOX DETECTED ERROR
8830 DMAB MBOX ERR
This bit is set if the MBox detects a parity error or NXM on the transfer of command/address from the DMAB transaction buffer. This bit cannot be set if bits <10> or <09> have been previously set. This bit is cleared by the CPU writing it. The DMAB Command/Address Register and DMAB ID Register will be locked if this bit sets. The setting of this bit will generate a local interrupt.

<07> MBZ
Must be zero.

<06> DMAA TRANSACTION BUFFER SBIA DETECTED A/D PARITY ERROR
8830 DMAA A/D ERROR
This bit is set for a data parity error when the read data is being transferred from transaction buffer A to the SBI during a DMA read. This bit cannot be set if bits <05> or <04> have been previously set. This bit is cleared by the CPU writing it. The DMAA Command/Address Register and DMAA ID Register will be locked if this bit sets. The setting of this bit will generate a local interrupt.

<05> DMAA TRANSACTION BUFFER SBIA DETECTED CONTROL PARITY ERROR
8830 DMAA CONTRL ERR
This bit is set for a control parity error when the read data is being transferred from transaction buffer A to the SBI during a DMA read. This bit cannot be set if bits <06> or <04> have been previously set. This bit is cleared by the CPU writing it. The DMAA Command/Address Register and DMAA ID Register will be locked if this bit sets. The setting of this bit will generate a local interrupt.

<04> DMAA TRANSACTION BUFFER MBOX DETECTED ERROR
8830 DMAA MBOX ERR
This bit is set if the MBox detects a pabity error or NXM on the transfer of command/address from the DMAA transaction buffer. This bit cannot be set if bits <06> or <05> have been previously set. This bit is cleared by the CPU writing it. The DMAA Command/Address Register and DMAA ID Register will be locked if this bit sets. The setting of this bit will generate a local interrupt.

<03> DMAI TRANSACTION BUFFER INTERLOCK TIMEOUT
8830 DMAI TIMEOUT
This bit is set if an interlock write masked does not occur within 512 SBI cycles (102.4 microseconds) after an interlock read masked. This bit cannot be set if bits <02>, <01>, or <00> have been previously set. The setting of this bit will generate a local interrupt. This bit is cleared by the CPU writing it. The DMAI Command/Address and DMAI ID Register will be locked if this bit sets.

<02> DMAI TRANSACTION BUFFER SBIA DETECTED A/D PARITY ERROR
8830 DMAI A/D ERROR
This bit is set for a data parity error when the read data is being transferred from transaction buffer I to the SBI during a DMA interlock read. This bit cannot be set if bits <03>, <01>, or <00> have been previously set. This bit is cleared by the CPU writing it. The DMAI Command/Address and DMAI ID Register will be locked if this bit sets. The setting of this bit will generate a local interrupt.

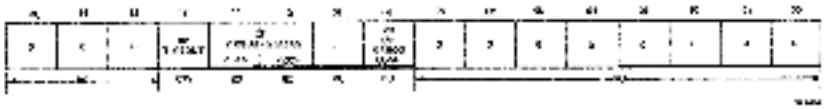




<01> DMAI TRANSACTION BUFFER SBIA DETECTED CONTROL PARITY ERROR
8830 DMAI CNTRL ERROR
This bit is set for a control parity error when the read data is being transferred from transaction buffer I to the SBI during a DMA interlock read. This bit cannot be set if bits <03:02> or <00> have been previously set. This bit is cleared by the CPU writing it. The DMAI Command/Address Register and DMAI ID Register will be locked if this bit sets. The setting of this bit will generate a local interrupt.

<00> DMAI TRANSACTION BUFFER MBOX DETECTED ERROR
8830 DMAI MBOX ERR
This bit is set if the MBox detects a parity error or RAX on the transfer of command/address from the DMAI transaction buffer. This bit cannot be set if bits <03>, <02>, or <01> have been previously set. This bit is cleared by the CPU writing it. The DMAI Command/Address Register and DMAI ID Register will be locked if this bit sets. The setting of this bit will generate a local interrupt.





The SMI error register stores information about CPU transactions which failed on the SBI due to timeout or error confirmation.

<31:16> ZERO (8836)
These bits are read as zeros provided by the zero fill logic.

<15:13> ZERO (8832)
These read only bits are forced to zero by hardware ground potential.

<12> CP TIMEOUT
BUS REG D<12> (S832)
This bit will be set when there is an SBI timeout on a CPU reference for one of the following reasons:

1. Unsuccessful access: When the SBIA does not receive an acknowledge confirmation for a CPU command/address or write data within 512 SBI cycles (102.4 microseconds) from the time the SBIA first requests the SBI. Unsuccessful access can be caused by the following:

   a. SBIA is unable to win the SBI through bus arbitration
   b. Target NEXUS is always busy when accessed
   c. The address is for a non-existent device or address
   d. Combinations of the first two

2. If the SBIA does not receive the read data within 512 SBI cycles of the acknowledge for the command/address, it is a read data timeout.

When this bit sets, Error Summary Register <23> will be set, locking Error Summary Register <31:26> (sync or reference), SBI Error Register <11:10, 08>, and the Timeout Address Register (referenced address). This bit is reset when the CPU writes it to a one. This will also reset <11:10, 08>.

<11:10> CP TIMEOUT STATUS <01:00>
BUS REG D <11:10> (S832)
The timeout status bits are made up of two signals as follows:

1. Bit <01>: Grace machine is in the read pending state.
2. Bit <00>: SBI confirmation equals 01, busy.

Together these two signals indicate the type of SBI timeout.

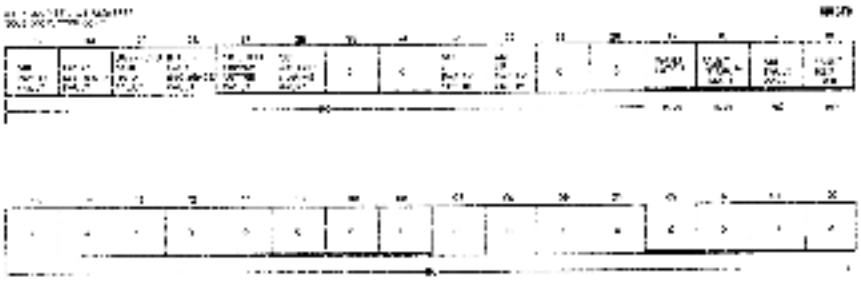
   a. 00: SBI NEXUS did not respond (No Response)
   b. 01: Device was busy (Busy)
   c. 10: Waiting for read data
   d. 11: Cannot happen

These bits are locked by Error Summary Register <23>, and reset when the CPU writes SBI Error Register <17>.

<09> ZERO (S832)
These read only bits are forced to zero by hardware ground potential.

<08> CPU SBI ERROR CONFIRMATION
BUS REG D <08> (S832)
This bit is set when the SBI state machine enters the error abort state if the SBI NEXUS has returned an error confirmation on a CPU read/write command/address cycle. If this bit is set, Error Summary Register <23> will be set, locking the Timeout Address Register, Error Summary Register <31:26>, and bits <12:10> of this register. This bit is reset when the CPU writes bit <12>.

<07:00> ZERO (S832)
These read only bits are forced to zero by hardware ground potential.



The SBI fault/status register saves information on SBI faults on transactions in which the SBIA may or may not have been involved. All SBI devices monitor the SBI every cycle and check for errors. If an error is detected, the fault wire is asserted and a fault/status register (part of the configuration register for non-CPU devices) is locked.

<31> SBI PARITY FAULT
SBI FAULT REG R <31>
This bit will set if the SBIA detects an SBI parity error on received SBI information. Bits <23:22> will indicate whether it was an address/data or control parity error. The register is written every SBI cycle, and locked when SBI FAULT is asserted. This bit will be cleared when SBI FAULT is deasserted. This bit is only valid if bit <19> is set.

<30> WRITE SEQUENCE FAULT
SBI FAULT REG R <30>
This bit is set if the SBIA is expecting write data, and receives SBI information, with no parity error, but the tag does not indicate write data (101). This bit is also locked when SBI FAULT is asserted, and clears when SBI FAULT clears. This bit is only valid if bit <19> is set.

<29> UNEXPECTED READ DATA
SBI FAULT REG R <29>
This bit sets if the SBIA receives information with the SBIA ID (100000) with a read data tag (000), by the SBIA is not expecting read data (no read pending). This bit is locked when SBI FAULT is asserted, and clears when SBI FAULT clears. This bit is only valid if bit <19> is set.

<28> INTERLOCK SEQUENCE FAULT
SBI FAULT REG R <28>
This bit is set if the SBIA receives a valid command/address for an interlock write masked but the interlock flip-flop is not set (an interlock read has not occurred). This bit is also locked when SBI FAULT is asserted, and clears when SBI FAULT clears. This bit is only valid if bit <19> is set.

<27> MULTIPLE TRANSMITTER FAULT
SBI FAULT REG R <27>
This bit will set if the SBIA detects an ID that is not the same as the ID it transmitted on the SBI. This bit is also locked when SBI FAULT is asserted, and clears when SBI FAULT clears. This bit is only valid if bit <19> is set.

<26> SBI TRANSMITTER DURING FAULT
SBI1 FAULT REG B <26>
This bit asks when the SBIA was the nexus transmitting on the SBI. This bit is locked when SBI FAULT is asserted, and clears when SBI FAULT clears. This bit is only valid if bit <19> is set.

<25:24> MBZ (SBI3)
Must be zero.

<23> SBI PI PARITY ERROR
SBI1 FAULT REG B <23>
This bit indicates that an SBI parity error was over SBI a <11:00>. It is only valid if bit <19> is set. It is locked when SBI FAULT is asserted, and will clear when SBI FAULT clears.

<22> SBI PO PARITY ERROR
SBI1 FAULT REG B <22>
This bit indicates that an SBI parity error was over SBI TAG <02:00>, SBI ID <04:03>, and SBI MASK <08:05>. It too, is only valid if bit <19> is set, and is locked when SBI FAULT is set. It will clear when SBI FAULT clears.

<21:20> MBZ (SBI3)
Must be zero.

<19> FAULT LATCH
SBIA REG B <19> (SBI3)
If an SBI xcxxx, including the SBIA, detects an SBI fault, the nexus will assert SBI FAULT. The SBIA, upon recognition of SBI FAULT will set the fault latch, which will keep SBI FAULT asserted. It will remain asserted until the CPU clears the fault latch by writing a one to bit <19>. SBI FAULT will be asserted for the following SBI error conditions:

1. Interlock sequence fault
2. Unexpected read fault
3. Write sequence fault
4. Multiple transmitter fault
5. Parity fault

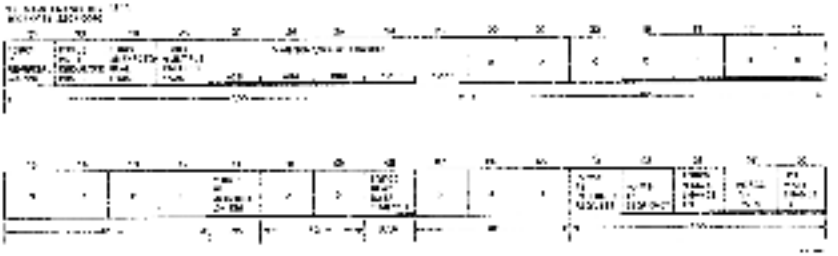
When this bit sets, Fault/Status Register <31:26> and <23:22> will be locked.

<18> FAULT INTERRUPT ENABLE
SBI1 FAULT INTR ENA
The CPU will set this bit by writing bit <17> to enable an SBI fault to generate an interrupt. The interrupt will be asserted if the fault latch, bit <19>, is set.

<17> SBI FAULT WIRE
SBI3 BUS REG D <17>
This bit indicates the state of the SBI FAULT signal.

<16> FAULT SILO LOCK
SBI3 BUS REG D <16>
This bit will be set when the silo locks due to an SBI fault. It will be reset when the CPU resets the fault latch, bit <19>.

<15:00> MBZ (SBI6)
Must be zero.



This register is used as a diagnostic and maintenance tool. Operational software does not use this register.

<31> FORCE PO REVERSAL
R/W:24 EXL: PO REV ON SBI
When set, this bit will cause the SBIA to transmit bad PO parity on the SBI for all SBIA to SBI transactions. This includes CPU read/write and DMA read data.

<30> FORCE WRITE SEQUENCE FAULT
R/W:24 FORCE SEQ FAULT
When set, this bit will force SBI TAG <01> to a logic 1. When used with a CPU write to an SBI nexus register, it will force the write data tag to 111, the diagnostic tag. This will cause a write sequence fault because SBI devices are looking for a tag of 101, write data.

<29> FORCE UNEXPECTED READ FAULT
R/W:24 FORCE UNEXP READ
When this bit is set, the maintenance ID, bits <27:23>, with a TAG of zero, will be repeatedly transmitted on the SBI (the data is undefined). When the nexus, as selected by the maintenance ID, receives read data (TAG = 0), it should assert NXM SBI FAULT because of the unexpected read data.

<28> FORCE MULTIPLE TRANSMITTER FAULT
R/W:24 FORCE MULT
Setting this bit forces a multiple transmitter fault in any selected nexus. The CPU will load the maintenance ID with the ID of the selected nexus, then read that nexus configuration register. On the cycle after the command/address is transmitted on the SBI, the SBIA will enable the SBI to continually transmit a TAG = 111 with the maintenance ID (data is undefined).

When the nexus transmits the read data, the ID transmitted by the nexus is the SBIA's ID. It will be ORed with the maintenance ID, and as long as the bits are not masked, will cause the nexus to detect a multiple transmitter fault.

<27:23> MAINTENANCE ID <04:00>
R/W:24 MAINT ID <04:00>
These bits are used to generate the maintenance ID in the following instances:

1. Generation of unexpected read fault
2. Generation of multiple transmitter fault
3. Used by the SBI as the compare ID
4. Used to check ID logic

<22:12> MBZ (8933, 8836, 8832)
Must be zero.

<11> FORCE P1 REVERSAL OF SBI
8821 FRC P1 REV ON SBI
When set, this bit will cause the SBIA to transmit bad P1 parity on the SBI for all SBIA to SBI transactions. This includes CPU read/write and DMA read data.

<10:09> MBZ (8832)
Must be zero.

<08> FORCE READ DATA TIMEOUT
8824 FORCE TIMEOUT
This bit will preset the state machine timeout counter to all ones when the state machine enters the read wait start state. The timer will expire on the first count, generating a timeout condition while waiting for CPU read data.

<07:05> MBZ (8812)
Must be zero.

<04> FORCE SBI INTERRUPT REQUEST
8824 MAINT REQ LVL
When set, this bit will enable SBI silo Comparator Register <07:04> and <01> to force interrupt requests and ASSERT on the SBI.

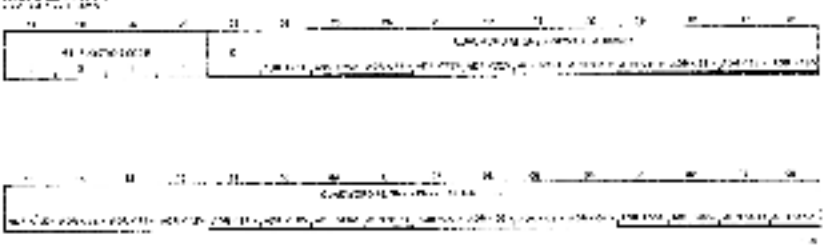
<03> FORCE IR REQUEST
8824 FORCE IR REQ
This bit will enable SBI Silo Comparator Register <15:00> to assert IR <15:00> on the SBI when a CPU Command/Address is transmitted. It is used in conjunction with a loop back read to test the SBIA and SBI nexus arbitration logic.

<02> FORCE MAINTENANCE IR
8824 FORCE MAINT IR
This bit will unconditionally assert the IR corresponding to SBI Silo Comparator Register <15:00>.

<01> FORCE INTERRUPT SUMMARY READ DATA
8824 FORCE ISR DATA
This bit is used to enable the SBIA to respond to an interrupt summary read to check out the circuitry that prioritizes the ISR data and generates the vectors.

During the response cycle of an interrupt summary read, the SBIA will enable the write data latch to be transmitted on the SBI along with a PAR, MASK, and ID of zero.

<00> USE MAINTENANCE ID
8824 USE MAINT ID
This bit will enable the use of the maintenance ID, bits <27:21> for diagnostic purposes.

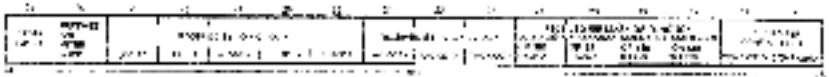


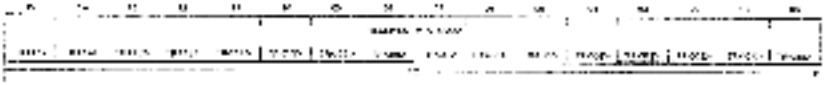
The purpose of the quadclear register is to clear ECC errors in SBI memory. The quadclear register does not exist as a physical register. When the CPU writes the Sbi Quadclear Register, the CPU register address, in the command/address, is decoded as a quadclear operation. The write data is used to generate the SBI command/address. Two longwords of all zero data is supplied by the SBIA.

<31:28> SBI FUNCTION CODE
Bits <31:28> must be 1011, as they will become the SBI function code, extended write masked. If this register is read, the quadclear is not done, and the read data will be all zeros supplied by the zero fill logic.

<27> MBZ
Must be zero. This bit is written as zero because the address must be a memory address, not an I/O address. This bit is also read as a zero provided by the zero fill logic.

<26:00> QUADWORD ALIGNED PHYSICAL ADDRESS - ADR <26:00>
These bits will become the quadword address in the command/address. This bit is also read as a zero provided by the zero fill logic.





The SBI silo is a read only register file that provides temporary storage of various SBI signals for the last 16 SBI cycles.

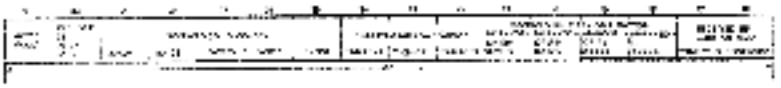
The assertion of SBI FAULT by any SBI nexus locks the silo, sets FAULT SILO LOCK in the SBI FAULT/STATUS register, and makes the data available through the SILO register. The silo may also be locked using the SBI COMPARE register. See DB86 SBIA Technical Description, EK-DB86X-TD for a description of the ATTN. An example of silo interpretation follows the register description.

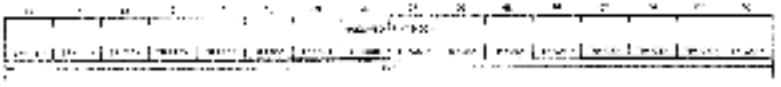
<31> AFTER FAULT
SUS REG D<31> (SB20)
SBI silo bit <31> is loaded with AFTER FAULT, an indication that the SBI fault condition has cleared. AFTER FAULT is only asserted for one SBI cycle, and will be written into the first silo location after the fault clears. It may be used to recognize frequently occurring fault conditions.

<30> RECEIVER SBI INTERLOCK
BUS REG D<30> (SB20)
Silo bit <30> is loaded with REC SBI INTLK from the SBI transceivers.

<29:25> RECEIVED SBI ID <04:00>
BUS REG D<29:25> (SB20)
Silo bits <29:25> are loaded with REC SBI ID <04:00>, an indication of which nexus is in control of the SBI. See the table that follows.

ID CODE	NEXUS	TR
--	-	0*
--	SBIA (EK4)	1**
16	SBIA (CP0)	2***
3	[illegible]	3
4	DW780	4
5	[illegible]	5
6	DW790	6
7		7
8	KA780	8
9	KI780	9
10	KA730	10
11	DW750	11
12		12
13	CI780	13
14	DW700	14
15		15
--		16





Notes:
-A TO 0 is used to hold the SBI for the next SBI cycle.
-A TO 1 is used by the SBIA for DMA reads to transmit the read data return word to the SBI. The ID will be the ID of the device that originated the DMA read transaction.
-A TO 2 is used by the SBIA for CPU read/write of SBI nexus registers.

<24:22> RECEIVED SBI TAG <02:00>
SUS REG D<24:22> (SS20)
Silo bits <24:22> are loaded with RCV SBI TAG <02:00>, an indication of the type of SBI cycle as follows:

1. 000: Read Data

2. 011: Command/Address

3. 101: Write Data

4. 110: Interrupt Summary Read

5. 111: Diagnostic Tag

<21:18> RECEIVED SBI MASK OR FUNCTION
BUS REG D <21:18> (SS20)
The contents of silo bits <21:18> depend upon the SBI tag. If the tag is 011, command/address, the silo is loaded with the bar function from bits <31:28>. Otherwise, the silo is loaded with the mask bits. The function codes are decoded as:

1. 0001: Read Masked

2. 0010: Write Masked

3. 0100: Interlock Read Masked

4. 0101: Interlock Write Masked

5. 1000: Extended Read

6. 1011: Extended Write Masked

The mask bit meanings for a command/address or write data are:

1. Mask <03> = 1; Read or write to byte 3
2. Mask <02> = 1; Read or write to byte 2
3. Mask <01> = 1; Read or write to byte 1
4. Mask <00> = 1; Read or write to byte 0

The mask bit meanings for read data are:

1. 0000: Valid read data
2. 0001: Corrected read data
3. 0010: Uncorrectable error

<17:16> RECEIVED SBI CONFIRMATION <01:00>
BUS REG D<17:16> (8820)
Silo bits <17:16> are loaded with the SBI confirmation bits which have the following meanings:

1. 00: No response
2. 01: Acknowledge
3. 10: Busy
4. 11: Error on command/address

<15:00> RECEIVED SBI TRANSFER REQUEST <15:00>
BUS REG D<15:00> (8819)
Silo bits <15:00> are loaded with any SBI transfer requests that may be asserted.

Example of interpreting a deposit byte to D#780 40 at TR3.

D/B 2E006000 AA

The SILO may be read as follows:

REPEAT 16 EXAMINE 2E006030

The following information would appear in the silo if it was looked after 16 SBI cycles (or less) by ART PANIC or by using the SILO COMPARE register. Note that the first silo location read would be the first location loaded. Only silo 4 locations are shown.

SILO CONTENTS	BREAKDOWN
90C00001	ID = 16, CPU; TAG = 3, C/S; FUNCTION = 0010, WRITE MASKED; TR = 0, IDLE
21440003	ID = 16, CPU; TAG = 5, WRITE DATA, MASK = 0011, WRITE BYTE 0
00010000	COMP = 01, ACK
90010000	COMP = 01, ACK



<31> COMPARATOR SILO LOCK
SS27 CMP SILO LOCK
The Comparator Silo Lock bit is set if the count in the silo counter has reached F. When this bit sets, the CPU will be interrupted if bit <30> is set. This bit is cleared when the CPU loads the silo count field with a count other than F.

<30> SILO LOCK INTERRUPT ENABLE
SS25 SILO LOCK INTR EN
The CPU sets this bit to enable an interrupt when bit <31>, CMP SILO LOCK, sets.

<29> LOCK UNCONDITIONAL
SS25 LOCK UNCOND
When this bit is set, the silo counter will count on each SBI cycle. It will cause a silo lock within 16 SBI cycles, depending upon the count that had been previously loaded into the silo count field.

<28:27> CONDITIONAL LOCK CODE <01:00>
SS25 COND LOCK CODE <01:00>
These two bits determine the comparisons that will enable counting the silo counter to achieve a silo lock. If the SBI data matches the silo comparator bits, for the enabled comparison, the counter is incremented. The conditions are as follows:

1. 00: No compare (no comparison is made)

2. 01: SBI ID

3. 10: SBI ID and SBI TAG

4. 11: SBI ID and SBI TAG and SBI COMMAND/MASK

<26:23> COMPARATOR COMMAND/MASK
SS25 CMP CMD/MSK <03:00>
These bits provide the base for the silo comparison, when it is enabled to compare the command/mask. If the SBI tag is 011, command/address, then this field is compared with SBI A <31:28>, the SBI function. If the SBI tag is other than 011, this field is compared to the SBI mask bits.

<22:20> COMPARATOR TAG
8225 CMP TAG <02:00>
These bits provide the base for the silo comparison, when it is enabled to compare the tag. This field is compared with SBI TAG <02:00>.

<19:16> COUNT FIELD
8219 COUNT FIELD
The CPU loads the silo counter with the two's complement of the number of SBI cycles to be loaded into the silo after a comparison is made. When the count reaches F, the silo is locked. The counter is also enabled by the Lock Unconditional bit <29>.

<15:00> MAINTENANCE TRANSFER REQUEST <15:00>
8225 MAINT TR <15:00>
These bits provide the means to simulate SBI transfer requests, SBI interrupt requests, and SBI alert for diagnosing the interrupt logic and SBI priority arbitration logic. They also provide a means of testing the lower 16 bits of the silo. They are controlled by SBI Maintenance Register bits <04:02> as follows:

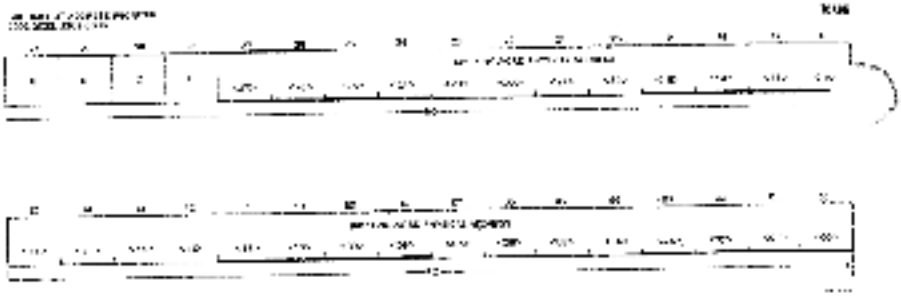
1. The asserted MAINT TR <07:04> bit will cause the corresponding SBI REQ <07:04> bit to be asserted if SBI Maintenance Register <04>, MAINT REQ ENA, is set.

2. If MAINT TR <04> and SBI Maintenance Register <04> are both set, SBI ALERT will be asserted.

3. If SBI Maintenance Register <03> is set, the asserted MAINT TR <15:00> will cause the corresponding SBI TR <15:00> to be asserted when a CPU command/address is transmitted on the SBI. See SBI Maintenance Register bit <03>.

4. MAINT TR <15:00> will cause the corresponding SBI TR <15:00> to be asserted if SBI Maintenance Register bit <02>, FORCE MAINT TR, is set.

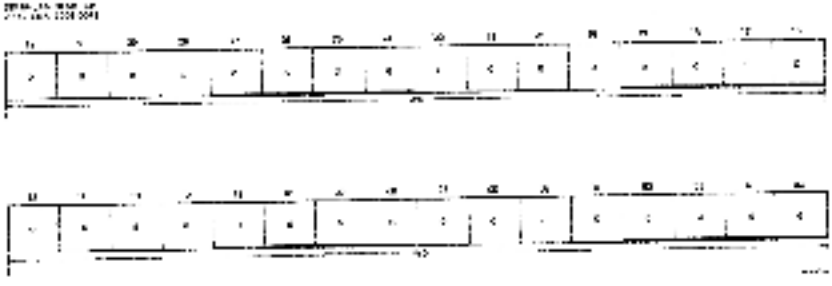
VAX 8600/8650 REGISTER DESCRIPTION
8913A SBI TIMEOUT ADDRESS
8913A SBI UNJAM



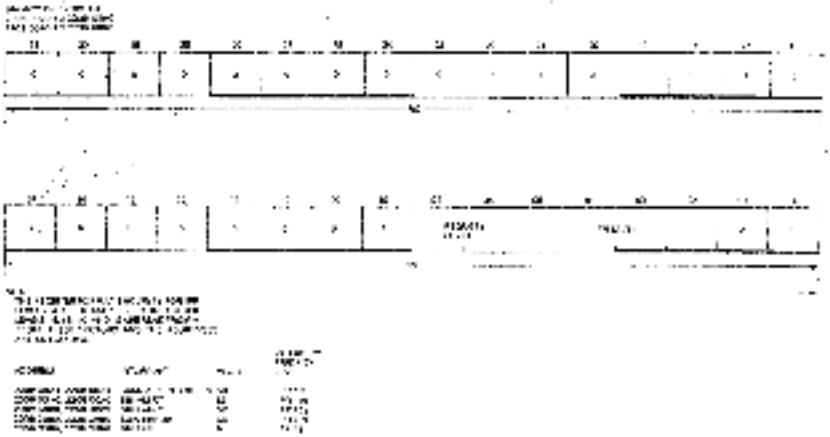
The SBI timeout address register holds the address for all CPU transactions in the SBIA. When an error occurs on a CPU transaction, the timeout address register is locked.

<31:28> MBZ (8926)
These bits are forced to zero by the zero fill logic.

<27:00> SBI LONGWORD PHYSICAL ADDRESS - BUS REG D <27:00> (8927)
The Timeout Address Register is loaded with the physical address, for the CPU command, each time a command/address is transferred from the file data latch to the command/address latch. It will be locked if Error Summary Register bit <7> is set.



<31:00> SBAQ
The SBI Unjam register does not exist as a hardware register. When the SBIA decodes the address of the Unjam register, for a CPU write, the Unjam sequence will be initiated. If this register is read, the contents will show all zeros, provided by the zero fill logic on 8926. For a read, the Unjam sequence will not be done.



The CPU will read the appropriate vector register in response to the arbitrated interrupt priority requests, which it will use, along with the I/O adapter number to build the vector register. If the interrupt being serviced originated on the SBI, the vector is made up of the interrupt priority request level and the IPR _level. If the interrupt being serviced originated on the SBIA, the vector is read from a 32 x 8 PROM.

<31:12> MBZ (SS36)
Must be zero.

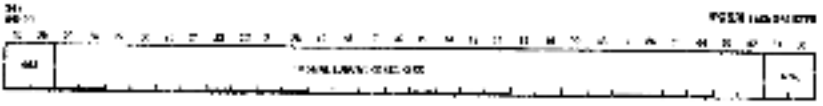
<11:09> MBZ (8830 or 8831)
Must be zero. If the interrupt being serviced originated on the SBI, these zeros are forced by ground potentials in the hardware on 8831. If the interrupt is a local SBIA interrupt, these zeros are provided by the zero fill logic.

<08> LOGIC ONE (8831)
SBI BUS REG <08>
This bit is tied to +3 volts for an SBI interrupt and is always read as a logic zero from the PROM.

<07:06> REQUEST LEVEL
SBI BUS REG <07:06>
These fields are supplied by the two least significant address bits, which correspond to the request level, for an SBI interrupt. They are provided by the PROM for a local SBIA interrupt.

<05:02> IPL LEVEL (SS91)
BUS BMG <05:02>
The IPL level is represented by bits <05:02> if the interrupt is an SBI interrupt. If the interrupt is a local SBIA interrupt, these bits are provided by the PROM.

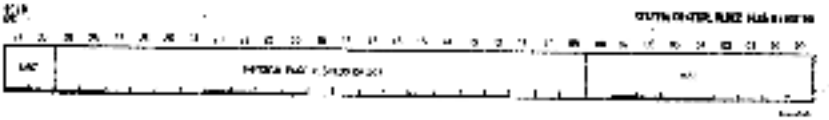
<01:00> ZERO
SBI BUS REG <01:00>
This field is read as zeros. It is supplied by the PROM for local SBIA interrupts, or inverting +3 volts for an SBI interrupt.



<31:30> MBZ
Must be zero.

<29:02> PHYSICAL LONGWORD ADDRESS
The address points to the first PTE in the SPT. In turn, this PTE maps the first page of System Space, that is, virtual byte address 80000000 (hex).

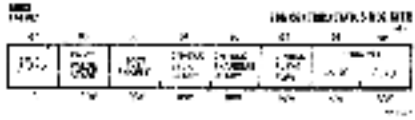
<01,00> MBZ
Must be zero.



<31:30> MBZ
Must be zero.

<29:09> PHYSICAL PAGE ADDRESS OF SYSTEM CONTROL BLOCK
The SCBA is a privileged register containing the physical address of the System Control Block, which must be page-aligned.

<08:00> MBZ
Must be zero.



<07> STOP CLOCK
When set, it indicates to the T-11 program that all SDR clocks have been generated and that the operation is complete.

<06> SHIFT VALUE START
The T-11 program asserts shift value start to generate the SDR clock or clocks needed to perform an SDR operation.

<05> -TEST READY
This is an unused pin, the input to DAL <5> via the SDR to the DAL multiplexer. It is used to inform the software that the console module is in the console test station.

<04> ENABLE STORE READY
Enable STR interrupts from the VAX CPU.

<01> ENABLE TRANSMIT READY
Enable TX interrupts from the VAX CPU.

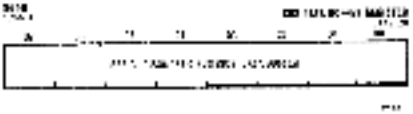
<02> SINGLE EVENT FLAG
When this bit is negated, eight SDR clocks are generated. When it is asserted, one SDR clock is generated.

<01:00> CHANNEL WRITE, READ
The direction of the data transfer is specified by setting either the CHANNEL WRITE or CHANNEL READ bits.

<03:00> SDCS
Specify the operations used to transfer data from a selected visibility channel. See table below.

CONTROL BIT SETTINGS TO READ OR WRITE A VISIBILITY CHANNEL

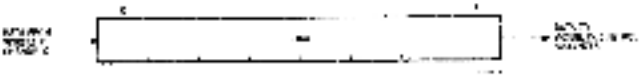
SINGLE EVENT	CHANNEL WRITE	CHANNEL READ	OPERATION
0	1	0	Shift eight bits in SDR out to visibility register (if SDMR VIS SHIFT = 1). Generates eight SDR clocks.
0	0	1	Shift eight bits from visibility register into SDR (if SDMR VIS SHIFT = 1). Generates eight SDR clocks.
1	1	0	Shift one bit in SDR out to visibility register (if VIS SHIFT = 1). Generates one SDR clock.
1	0	1	Shift one bit from visibility register (if VIS SHIFT = 1). Generates one SDR clock.
1	0	0	Load visibility register from diagnostic serial/shift (if VIS SHIFT = 0). Generates one SDR clock.



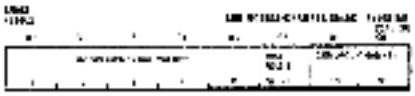
The SCDP is an 8-bit shift register. It is parallel loaded by the T-11 program with the data to be written out to a visibility or control channel.

This data is either diagnostic terminator select and enable information loaded prior to shifting data back to the console, or control data when the channel is used for control purposes.

The serial data emitted back to the console is loaded in the SCDP where it may be read by the T-11 Program. The direction of the shift paths is as follows:



For additional information, refer to the CPU section of the Maintenance Guide and the VAX 8600/8650 Console Technical Description, EK-KA866-TD.



The SCMS register is used by the console program to control/sec-up the SDB prior to initiating the SDB sequence via the SDCS register. For additional information pertaining to SDB operation and the register, refer to the SDB section of the Maintenance Guide and to the KA86 Technical Description Manual (EK-KA86C-TM).

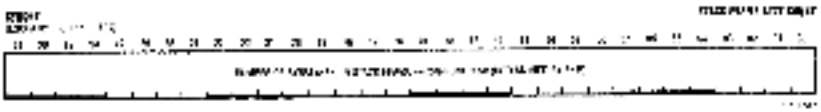
<0:03> MODULE/CHANNEL SELECT
The module/channel select field determines which channel to perform the SDB operation on and is encoded as shown in the following table.

Module Channel Select	Module	Control Channel (NOTE 1)	Module Channel Select	Module	Control Channel (NOTE 1)
00	VDA	Yes	16	CSB	Yes
01	VBP	Yes	17	CI4	No
02	MCD	No	20	IOA 0	Note 2
03	EBU	Yes	21	IOA 1	Note 2
04	IDP	No	22	IOA 2	Note 3
05	ICA	Yes	23	IOA 3	Note 3
06	ACS	No	24	NTN	No
07	CLK	Yes	25	Spare	--
10	BDP	Yes	26	Spare	--
11	EBX	Yes	27	Spare	--
12	MCC	Yes	30	SBI 0	N/A
13	MAP	No		Loopback	Yes
14	EBD	No	34	SCF	Yes
15	ESC	Yes			

NOTES: (1) Module has both control and visibility channels.
(2) IOA0 and IOA1 visibility and control is available to manufacturing only (via special test ADBus hookup)
(3) IOA2 and IOA3 are currently nonexistant

<02> VISIBILITY SHIFT
This signal is distributed (daisy chained) to each visibility channel and used to control the loading and shifting of shift register logic in the selected visibility channel. When asserted, the visibility channel shift register will shift data one bit position for each SDB clock. When cleared, the visibility channel shift register will perform a parallel load operation.

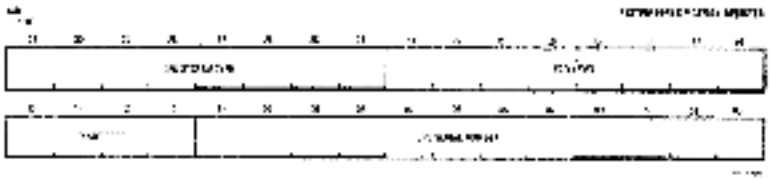
<01:00> CONTROL CHANNEL S2, S1
These two signals are distributed (daisy chained) to each control channel and are implemented within the control channel to perform specific functions. The implementation of these bits within the control channel is channel dependent (may be implemented differently in each channel). Refer to the appropriate technical description manual for the module (box) in question for a description of S0, S2 and S1 operation.



This longword is built by the Error Handling Microcode (EBK). It contains the number of bytes (in HEX) that the SBM will assemble in the CBox Scratch Pad RAMS (location 18 through 1D). When the SBM is finished building the Machine Check Stack Frame, it calls the Interrupt Exception Microcode. The microcode pushes the contents of the Scratch Pad RAMS onto the Interrupt Stack and calls the VMS Machine Check (MCHK) handler. The MCHK handler will use the contents of this longword to process the stack but the SPBCNT will not appear in the Stack Frame Record written by XBRERR.SYS.

<31:08> RESERVED

<07:00> BYTE COUNT
Indicates the number of bytes, currently in HEX-88 Decimal, that the SBM has pushed onto the Interrupt Stack.





The System Identification (SID) register contains information unique to the system: the CPU serial number, the processor type, the manufacturing plant name and the CPU revision. The source of this register is the Console, with bits <31:24> being generated by Console software and bits <23:00> jumper selectable on the CPU backplane, J09. The contents of this register (except the software generated bits <31:24>) are also addressable from the console, using the console registers as shown in the PIDMAP above: SID2 <23:16>, SID1 <15:08> and SID0 <07:00>.

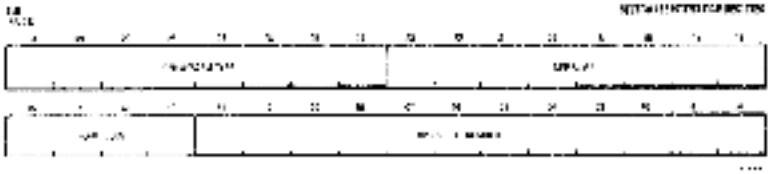
<31:24> CPU TYPE
This field identifies the CPU type and specifies the programmer as 04(X). These bits <31:24> are generated by the console software when the register is read (i.e., are not set in jumpers on the backplane like the rest of the bits in this register).

<23> VAX CPU TYPE
0 = VAX 8600, 1 = VAX 8650

<22:16> CPU REVISION
This field specifies the hardware revision of the kernal CPU. Only MAJOR ECO revision levels are considered here. To calculate the letter revision, simply use a one-to-one conversion from number to alphabet thus: C1(X) = A, C2(X) = B, C3(A) = C, etc.

<15:12> MANUFACTURING PLANT CODE
This field contains a code which represents the manufacturing plant where the CPU was built. Current codes are as follows:

PLANT CODE (HEX)	MANUFACTURING PLANT
1, 8, 9	GALWAY
5, 6, 7	FRANKLIN, MA
A, B, C	BURLINGTON, VT
D, E, F	MARLBORO, MA



<11:00> CPU SERIAL NUMBER
This field contains the CPU serial number (also shown on the silver label attached to the CPU cab: top left rear) converted to hexidecimal. For example, if the CPU serial number displayed on the serial number tag were XBU0617, the conversion would be 269(x).

Additional Notes:

(1) Refer to the diagram at left which illustrates the relative bit positions of the SID jumper, located on the CPU backplane.

(2) The SID register is read by VMS (via the Console) and will be logged into each entry of the error log. To read manually, you can use the MFPR instruction (IPR number 3E) or from the Console (hex):

>>>E SID

? 3E 04045260

In addition, the Console command 'SHOW VERSION' will display the Hardware Revision level in hex, which is the contents of SID <23:16> or console register SID2 / 176403.

(3) VAX 8650 System SID Jumper Settings

JP PINS	SID BIT	JUMPER (IN/OUT)	VALUE	FUNCTION
17 - 18	23	OUT	1	1 = 8650
19 - 20	22	IN	0	
21 - 22	21	IN	0	
23 - 24	20	IN	0	
25 - 26	19	IN	0	
27 - 28	18	IN	0	
29 - 30	17	IN	0	
31 - 32	16	OUT	1	1 = REV A

To verify the current SID register jumper settings, perform an EXAMINE SID at the VMB prompt (>>>) as shown in the following example:

>>>EXAMINE SID

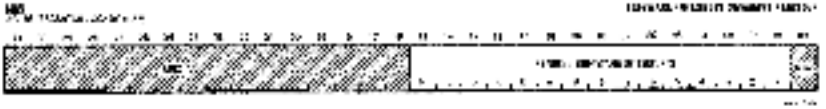
SID[0]55 [example system 1]



<31:04> MBZ
Must be zero.

<03:00> INTERRUPT REQUEST LEVEL
The software interrupt request register (SIRR) is a write-only four bit privileged register used for making software interrupt requests. Valid software interrupt levels are 0F:01 (interrupt levels 1F-10 are reserved for hardware).

Executing MTPR SIRR requests an interrupt at the level specified by bits <03:00>. Once a software interrupt request is made, it will be cleared by the hardware when the interrupt is taken. If SRC <03:00> is greater than the current IPL, the interrupt occurs before execution of the following instruction. If SRC <03:00> is less than or equal to the current IPL, the interrupt will be deferred until the IPL is lowered to less than SRC <03:00> with no higher interrupt level pending. This lowering of IPL is by either REI or by MTPR x, IPL. If SRC <03:00> is 0, no interrupt will occur.

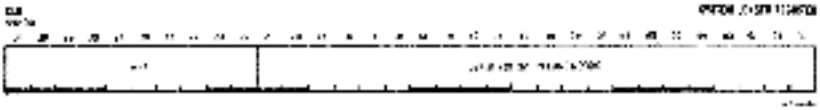


<31:16> MBZ
Must be zero.

<15:01> PENDING SOFTWARE INTERRUPTS - Levels 01-0F (RSX)
The processor provides 15 priority interrupt levels for use by software. Pending software interrupts are recorded in the Software Interrupt Summary Register (SISR). The SISR contains 1's in the bit positions corresponding to levels on which software interrupts are pending. All such levels, of course, must be lower than the current process IPL, or the processor would have taken the requested interrupt.

SISR is a read/write privileged register accessible only to privileged software. At bootstrap time, the contents of SISR is cleared. The SISR is accessed using MFPR and MTPR instructions.

<00> RESERVED



<31:22> MBZ
Must be zero.

<21:00> LENGTH OF SPT IN LONGWORDS
The System Virtual Address space is defined by the System Page Table (SPT), which is a collection of Page Table Entries (PTE's). The SPT is always located in physical address space. The base address of the SPT is also a physical address and is contained in the System Base Register (SBR). The size of the SPT in longwords (that is, the number of PTE's) is contained in the System Length Register (SLR).



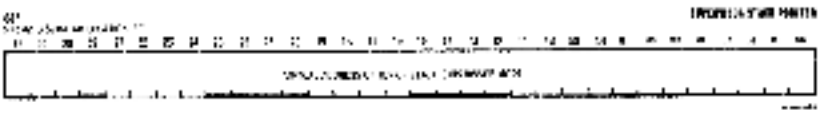
<07:00> CURRENT CONTENTS OF SPADR
When accessed by the Console, this register will contain the current contents of the Scratch Pad address lines <07:00>. Note that a CSM Overlay is required to access these contents, which may not reflect the actual address prior to loading the CSM Overlay. To examine the contents of SPADR before CSM is started, the Console must examine CSM.SPADDR (which is RSC location C2).

NOTE

1. The SPADR register (read/write) is accessed via the following console commands:

   >>>EXAM SPADR<cr>
   >>DEPOSIT SPADR data<cr>

2. The SPADR lines are generated on logic print MSPC by the MDD MCA and are distributed and used on the MDP module only and are not available on the backplane or via SDD visibility.



<31:00> SUPERVISOR STACK POINTER
Contains the stack pointer to be used when the current access mode field in the PSL is 1 (Supervisor Mode).



<31:00> CONTENTS OF STATE REGISTER
The STATE register is utilized by EBox Microcode as a miscellaneous register to control the microprogram flow. A microinstruction is used to load the STATE register from the ARBUS (refer to SSP.MIC and any EDP block diagram). The SEN field of the EBox microword is used to control microprogram flow and may be used to select the various bits of the STATE register to control selection of DPC bits <02:00>.

NOTE

1. The STATE register (read/write) is accessed via the following console commands:

   >>>EXAM STATE<cr>
   >>>DEPOSIT STATE data<cr>

2. This register is also available on the SSP visibility bus (all signals are generated on Print:EDPB and are remembered on the CSB mcon_m).

STATE BIT	SIGNAL NAME	EDS SYMBOL	S/P PIN
7	EDP STATE 7 H	V$E138	AC10B34
6	EDP STATE 6 H	V$E109	AC10A36
5	EDP STATE 5 H	V$E177	AC10B90
4	EDP STATE 4 H	V$E139	AC10A76
3	EDP STATE 3 H	V$E134	AC10B74
2	EDP STATE 2 H	V$E135	AC10A07
1	EDP STATE 1 H	V$E176	AC10A12
0	EDP STATE 0 H	V$E178	AC10A13



<31:24> STXCS0 STATUS CODE
Status of current transfer. Read only by CPU. Initialized to ONE by the console.

CODE	STATUS
01	Transaction_Complete
02	Continue_Transaction
03	Transaction_Aborted
04	Return_Device_Status
80	Hardware_Error_During_Transaction
81	Hardware_Error_During_Transaction

<23:16> STXCS2 HIGH ORDER DISK ADDRESS
Write only by CPU. Logical block number of current transfer. The console software will NOT perform bad block replacement on the RL02.

<15:08> STXCS1 LOW ORDER DISK ADDRESS

<07:00> STXCS0

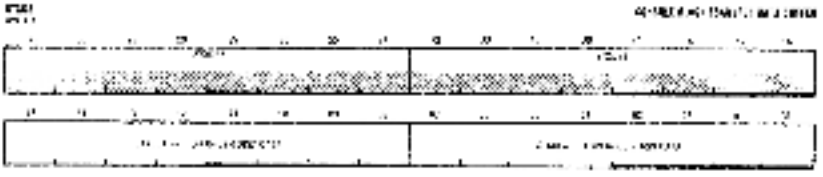
<07> READY
Read only by CPU. Initialized to ONE by the console.

<06> INTERRUPT ENABLE
Write only by CPU.

<05:04> RESERVED

<03:00> DISK FUNCTION CODE
Disk functions to perform. Write only by CPU. Initialized to ZERO by the console.

CODE	FUNCTION
0	No_Operation
2	Continue_Transaction
3	Abort_Current_Transfer
4	Read_Device_Status
5	Write_Block_Data
6	Read_Block_Data



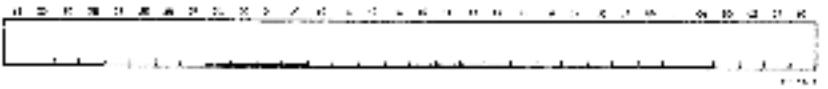
<31:16> STXDB1 Must be zero

<23:16> STXDB2 Must be zero

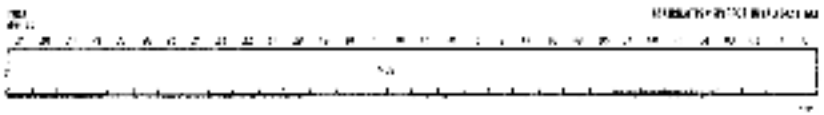
<15:08> STXDB1 High order disk data

<07,00> STXDB0 Low order disk data
Data should be read from the STXD8 only when STXC9 RDY, STXC8 <07>, is set, and is valid only if the STXC9 status field is 1 (Continue transaction) or 4 (Return Dev on Status), or the function field is 4 (Reset and Return Device Status).

Data is to be written in this field by the IOP only when the STXC9 status field is 2 (Continue Transaction) and the function field is 5 (Write Block Data).



The TXCHK register is included in the architecture due to anticipated use by VAX/VMS. Its specification and use is reserved to DIGITAL. On the VAX-11/730, the condition code V bit is cleared on a write to PXCHK if the virtual address has I/O space or if the protection field in the PTE says "no access for all modes."

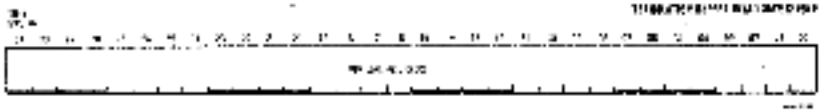


When the software changes a System Page Table Entry which maps any part of the current process page table, all process pages so mapped must be invalidated in the translation buffer. They may be invalidated by moving an address within each such page into the TBIS register. They may also be invalidated by clearing the entire translation buffer. This is done by moving 0 to the Translation Buffer Invalidate All (TBIA) register with the MTPR instruction.

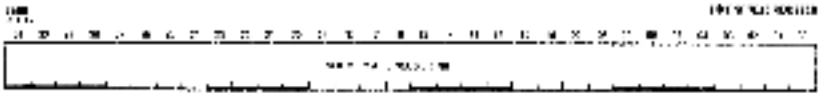
The translation buffer must not store invalid PTE's. Therefore, the software is not required to invalidate translation buffer entries when making changes for PTE's that are already invalid.

When the location or size of the system map is changed (SBR, SLR) the entire translation buffer must be cleared.

Whenever Memory Management Enable (MME) is a 0, the contents of the translation buffer are UNPREDICTABLE. Therefore, before enabling memory management at processor initialization time, or any other time, the entire translation buffer must be cleared.



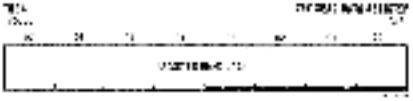
When the software changes any part of a valid Page Table Entry for the system or a current process region, it must do an MTPR a virtual address within the corresponding page to the translation buffer Invalidate Single (TBIS) register with the MTPR instruction.



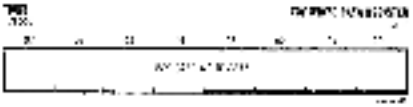
The Time of Year (TOY) clock is an elapsed time indicator for VMS and the means by which VMS knows whether the console is running. Power for the TOY clock (+5V DC) is supplied by the Battery Backup Unit (BBU) as long as the batteries can retain enough charge and if the TOY ON/OFF switch on the BBU is in the ON position. If the batteries are fully charged, the BBU can supply power to the TOY clock for 100 hours.

The TOY clock is implemented by a 32-bit counter that is incremented at a rate of 10 ms per count. It uses an Am9513 system timing control chip that is controlled by the "Tell" program. VMS communicates with the TOY clock via TODR. The VAX Time of Day Register. The X8600 implements this register as a software register in the CBus RAM. VMS loads and reads the TOY clock with MTPR/MFPR #TODR. The assembler translates "TODR" to address "1B" (hex). The SBIA subsystem in turn interprets "1B" as a CBus RAM location. If the TOY clock count becomes invalid, VMS asks the operator to load the correct time.

The timing of the Am9513 is controlled by a crystal which is also used by the console operating system for SNA communications. If the BBU +5V is lost, SNA communications as well as TOY synchronization cease.

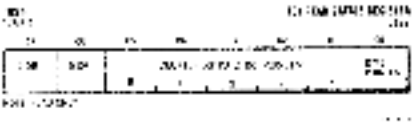


See TADR Register Description.



All TOY Chip (Am9513) registers (except the Command Write Register) can be read or written one byte at a time from the Read/Write Data Registers via the TOY Data Port. The Data Port is enabled when the C/-D input to the Am9513 (CLK0 XA1 ADR 2) is true.

The registers that can be accessed in the TOY module are: the Data Pointer and Master Mode Registers; and for each counter group, the Hold, Load and Counter Mode Registers. The Data Pointer register determines which register is accessed.



The Read Status Register (read-only) allows the byte pointer bit (BP) and the state of the courier outputs (OUT <5:0>) to be examined. The Read Status Register may also be used via the Data For...



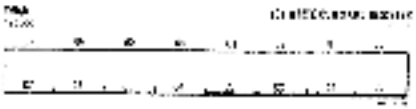
The TODR register is implemented in the CBus BAX. Refer to the 6400 Console Technical Description manual (EK-6ABEC-TM), for a detailed description of the clock operation.

<07> TOY SELECT
This bit has two uses:

1. It must be before (and cleared after) the console updates the BTOY Register. This will cause the XBox firmware to read the TOY register (instead of the BTOY Register) if it executes an MFPR TODR while the console is in the process of updating the BTOY Register.

2. It is set by the Data Strobe Auto (I updates the TODR Register (via an MTPR TODR). This will cause the Console to update the BTOY Register in the Console and the TOY Counter in the 8113 (TOY) Chip. The Console will perform the update request during its next incoming cycle.

<06:00> UPDATE ANOMALY COUNT
This field contains a count of the number of times that the BTOY register has been written (updated via a MTPR TODR instruction) by the XBox firmware.



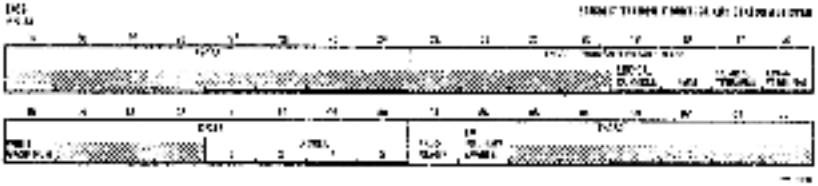
The 8513 (CSR Chip) Control Registers include the 8-bit Command and Status Registers which is accessed via the Control Port. The Control Port is enabled when the C/-D input to the 8530 (CSR3 bit 6OR 2) is false. The following table lists the CSR Chip Commands.

COMMAND CODE								
C7	C6	C5	C4	C3	C2	C1	C0	FUNCTION
0	0	0	D3	D1	D0	D2	D1	Load Data Pointer Register with contents of R and S fields. (000x(x)(1)0 or 0=111)
0	0	1	S5	S4	S3	S2	S1	Arm counting for all selected counters
0	1	0	S3	S6	S5	S7	S1	Load contents of specified source into all selected counters (NOTE 1).
0	1	1	S2	S6	S5	S2	S1	Load and arm all selected counters (NOTE 1).
1	0	0	S7	S4	S3	S2	S1	Disarm and save all selected counters (NOTE 1).
1	0	1	S5	S4	S3	S2	S1	Save all selected counters in Hold Register. (NOTE 1).
1	1	0	S5	S4	S3	S2	S1	Disarm all selected counters (NOTE 1).
1	1	1	0	1	N4	N2	N1	Set output bit for counter N (N=000 to 101).
1	1	1	0	0	N4	N2	N1	Clear output bit for counter N (N=000 to 101).
1	1	1	1	1	N4	N2	N1	Step counter N (N=000 to 101).
1	1	1	0	1	0	0	0	Disable Data pointer req., set MR<14> (NOTE 2).
1	1	1	0	1	1		0	Calc off FOUT, set MR<12> (NOTE 2).
1	1	1	0	1	1	1	1	Force 16-bit mode, set MR<11> (NOTE 2).
1	1	1	0	0	0	0	0	Enb. Data pointer req., clear MR<14> (NOTE 2).
1	1	1	0	0	1	1	0	Calc on FOUT, clear MR<12> (NOTE 2).
1	1	1	0	0	1	1	1	Enter 8-bit mode, clear MR<11> (NOTE 2).
1	1	1	1	1	0	0	0	Enable Prefetch for Write Operations
1	1	1	1	1	0	0	1	Disable Prefetch for Write Operations
1	1	1	1	1	1	1	1	Master reset.

NOTE

1. Source is determined for each counter by its Counter Mode Register.

2. Data or clears appropriate bit in Master Mode Register (MR).



<31:24> TXCS3
Must be zero

<23:16> TXCS2

<15:16> TRANSMIT ENABLE MASK
This field is Read/Write by the CPU and Read only by the Console. It is initialized to 01 by the Console (Local Line Enabled).

A bit set in this field indicates that the CPU wants to activate interrupts for that line. When the line is ready the console will generate an interrupt via the TXCS.

<15> ENABLE LOGICAL TERMINAL
Corresponds to the "LOGICAL" Console Subsystem (i.e., the Console to CPU interface path).

<14> ENABLE EVM
Corresponds to EVM data line.

<13> ENABLE REMOTE PORT
Corresponds to Console Remote Services Port.

<12> ENABLE CONSOLE TERMINAL
Corresponds to Console Terminal.

<11:08> TXCS1

<15> Write Mask Bow
Read/Write by CPU and Console. Initialized to zero by the Console.

This bit is set by the CPU to allow the MTPR instruction to write to the Transmit Enable Mask field. The bit is used by the Console to determine which operation caused the Console interrupt; a Mask Field Write or a TXDB Write.

<14:12> MBZ
Must be zero.

<11:08> ID Field
Read only by CPU. Initialized to zero by the Console.

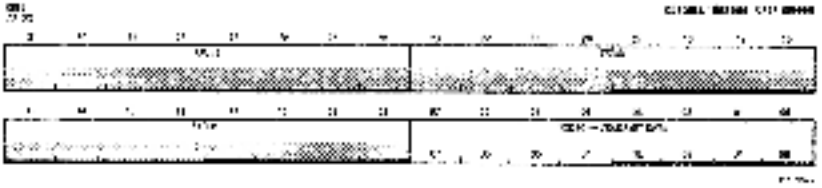
When the CPU receives a "RDY" interrupt, this field will identify the line that caused the interrupt, i.e., which line is now ready to transmit data.
ID = 0 Console Terminal
ID = 1 Remote Services Port
ID = 2 Environmental Monitoring Module (EMM) Data
ID = 3 Logical Console Data
ID = F No Lines Currently Enabled

<07:03> TXCS0

<07> TXCE Ready
Read only by CPU. Initialized to one by the Console. When set to a one, this bit indicates that the line specified in ID Field is ready to transmit, provided that the Transmit Enable Mask Field is not zero.

<06> TXCS Interrupt Enable
Read/Write by CPU. Initialized to zero by the Console. When the logical AND of "TXCS RDY" and "TXCS INTERRUPT ENABLE" is 1, a CPU interrupt is generated via System Control Block (SCB) vector "F0".

<05:03> MBZ
Must be zero.



<31:08> MBZ
Must be zero.

<07:00> Data Byte
Write only by CPU. Initialized to zero by the Console.

The code in TXDB <11:08> (ID Code) determines the destination of this data byte.

TXDB ID	TXDB DATA BYTE DESTINATION AND DEFINITION
0	Console Terminal. Output the Data Byte to the Console Terminal line (CTY).
1	Remote Services Port. Output the Data Byte to the Remote Terminal Line (RTY).
2	Environmental Monitoring Module (EMM). Output the Data Byte to the EMM. The TXDB data byte may contain any one of four HEX values. All other values will be ignored by the Console. 00 = CPU Request for EMM Status Information. The EMM data response is returned via the RX registers. 01 = CPU Request for System Environment Information. The EMM data response is returned via the RX registers. 10 = CPU Command to Control Regulator Margins. 10: Byte 1 = 0 -- Set all margins NORMAL Byte 1 = 1 -- Set all margins LOW Byte 1 = 2 -- Set all margins HIGH 11 = CPU Command to cancel current and Queued EMM Requests.
3	Logical Console. Send the Data Byte to the Logical Console for processing. The TXDB data byte may contain any one of the following HEX values. All other value will be ignored by the Console. NOTE The following codes (2, 3, and 4) return no data to the CPU. 02 = Request Console to initiate system. 03 = Request Console to clear the Console copy of the Warm_Start_Flag.

TXCS ID	TXDB DATA BITS DESTINATION AND DEFINITION

64 = Request Console to clear the Console copy of the Cold_Start_Flag.

NOTE

The following modes (0, 1, IV, 1), and 10) return data to the CPU via the RX registers.

10 = Request the value of the Console's copy of the Warm_Start_Flag.

11 = Request the value of the Console's copy of the Cold_Start_Flag.

12 = Request the 16-bit value of the system microcode version currently being executed.

13 = Request 4 bytes of nonvolatile array configuration data.

20 = Used by diagnostics and error handler verification tools. Commands the Console to begin accepting a Console Command String (CCS). Any previous CCS in progress is aborted.

NOTE

- The CCS may consist of up to 512 ASCII bytes including the Null Terminator.
- Bit <7> must be set in each ASCII byte that is part of the CCS. The Console uses Bit <7> to identify the data as part of the CCS.
- Individual commands in a CCS must be separated with a <cr><lf> character pair.
- The Console stores the ASCII bytes in a buffer until it detects a Null Byte. The Null Byte terminates CCS mode and causes the Console to begin processing the CCS. The Console aborts the CPU, exits CIO Mode, enters CIO Mode, processes the command (just as though it were entered via the CTI). The Console then exits CIO mode, enters PIC Mode, "CONTINUES" the CPU from its halted PC, and re-enters CCS mode.
- The Console will remain in CCS Mode until it receives a CCS terminator character set (<cr><lf><null>).



TXDB (1) TXDB DATA BYTE DESTINATION AND DEFINITION

CDS EXERUN - The Console will reject the CCD and return an error code via the RX registers if:

1. The CCD EXTENDED time SIZE byte SPECIFIED EXCEEDS

2. The front panel switch indicates that the CONSOLE is LOCKED. The check for the locked Console is not made until the KILL character is received.

Because of the possibility of an RX short message occurring at any time during the issuing of a CCD, the CPU program is required to poll the RX register during the transfer.

21 = Request the 32-bit value of the VAX PC at the time of its last halt.

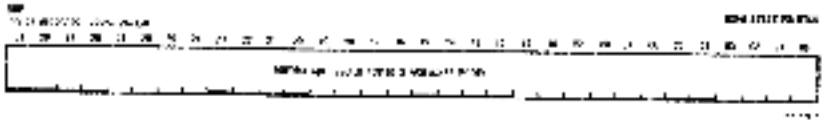
22 = Request the 32-bit value of the CSM Status at the time of its last entry.

30 = Request to the console to have it return the status of the two machine SNAPSHOT files.

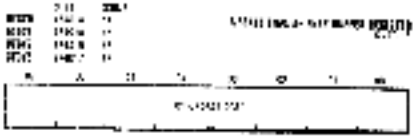
31 = A request to the console to have it invalidate the SNAP1.DAT file. The CPU should make this request after successfully processing the SNAP1.DAT file.

3A = A request to the console to have it invalidate the SNAP2.DAT file. The CPU should make this request after successfully processing the SNAP2.DAT file.

70 = Cancel all current and Queued Logical Console Requests.

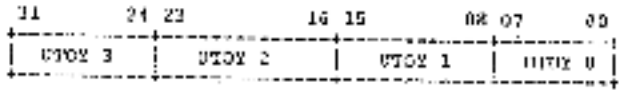


<31:00> USER STACK POINTER
Contains the stack pointer to be used when the current access mode field in the PSL is 3 (User Mode).



The UTOY register consists of four CBus RAM byte locations.

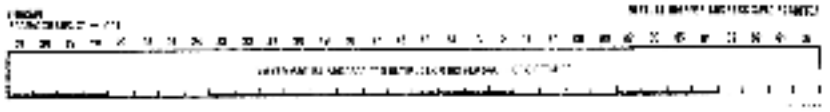
<07:00> TOY UPDATE DATA
The EBox microcode will load the UTOY registers with the 32 bit data specified in an MTPR #IODR instruction. The byte alignment is as follows:



When the CPUx executes a MFPR #IODx instruction, the EBox microcode will read the data from the UTOY registers if the UTOY registers are being updated (see STOY register description).

NOTE

Refer to the KA86 Console Technical Description manual (EK-KA86C-TD), for more detailed information on TOY clock operation.



<31:00> IBOX IDUX PORT VIRTUAL ADDRESS
This register contains the last buffer virtual address that was acknowledged by the MBOX (PA ACK).



<31:03> RESERVED

<02:00> VALID PC BITS
These bits represent the validity of the three Program Counters: CPC, IGA and PGA. If the valid bit is set, then the corresponding register contains valid data. When the CSP overlay is called to read this register, it simply reads the status of the following signals and sets the corresponding bits:

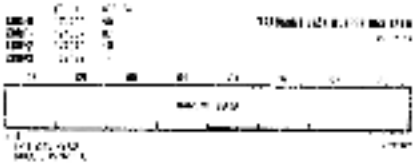
VPC B.T	SIGNAL NAME	MUX SIGNAL	ICB B/P PIN
2	ICB CPC VALID H	VSK144	AC12C24
1	ICB IGA VALID H	VSE161	AC12B56
0	ICB PGA VALID H	VSE129	AC12C14

NOTE

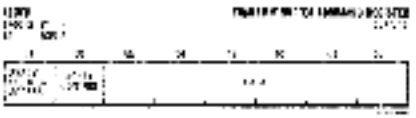
1. These signals are generated on the TCB module (prints ICB8, ICB9) and are well terminated on the ICB module.

2. The VPCBITS register (read only) is accessed via the following console command:

>>>EXAM VPCBITS



These four registers (XBUF0-3) are used in conjunction with the XBUFC register for transferring data packets from the EBox microcode to the console. This communication occurs only in Console I/O (CIO) mode and will take place between CSM and MCP (if in MACRO context) or DSM and DCP (if in DIAGNOSTIC context). Refer to CSM and DSM specifications, and the "KA86 Console Technical Description" manual, EK-KA86C-TD for operation of these registers.



The XBUFC register is used in conjunction with the XBUF0-3 registers for communication between the console software and EBox microcode. This communication occurs only in Console I/O (CIO) mode and will take place between CSM and MCP (if in MACRO context) or DSM and DCP (if in DIAGNOSTIC context). Refer to CSM and DSM specifications, "KA86 Console Technical Description", EK-KA86C-TD, and "VAX 8600/8650 System Diagnostic User's Guide", EK-KA86D-UG.

<07> READY RIGHT-OF-ACCESS BIT
When set indicates that the EBox microcode may load data into the XBUF registers. When cleared indicates that the Console software may read the XBUF registers.

<06> PACKET CONTROL
When set indicates that another packet will follow. When clear indicates that the current packet is the last one for this transaction.

<05:00> DATA
Application dependent; refer to both the MCP and CSM specifications.