EK-CIBCA-UG-001

CIBCA User Guide

digital™

EK-CIBCA-UG-001

CIBCA User Guide

Prepared by Educational Services
of
Digital Equipment Corporation

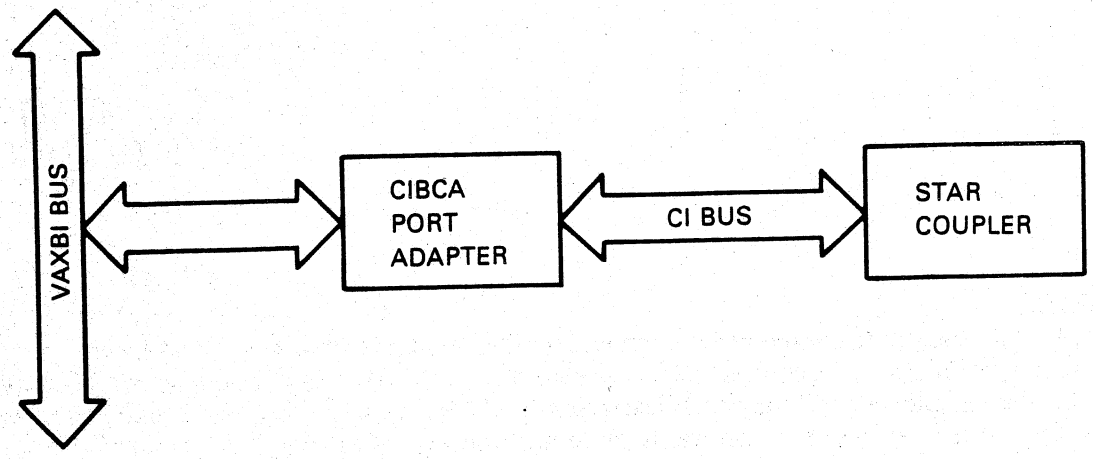


Figure 1-1 Simplified CIBCA Port Adapter Connection

1.2.2 Features

- VAX Backplane Interconnect design
- Diagnostic data loopback (internal/external) capability
- Data integrity via cyclic redundancy checking
- Round-robin arbitration at heavy loading
- Contention arbitration at light loading
- Packet-oriented data transmission
- Immediate acknowledgment of packet reception
- Operational modes
  - Disabled
  - Enabled
  - Uninitialized
- Dual signal paths

1.3 SPECIFICATIONS

CI GENERAL SPECIFICATIONS

Priority arbitration	
Light loading	Contention
Heavy loading	Round-robin
Parity	Cyclic redundancy check
Data format	Manchester-encoded serial packet

ENVIRONMENTAL SPECIFICATIONS

Temperature	
Operating	10°C to 40°C (50°F to 104°F) ambient temperature with a gradient of 10°C (18°F)/hr
Storage/shipping	−40°C to 70°C (−40°F to 158°F) ambient temperature with a gradient of 2°C (36°F)/hr
Relative Humidity	
Operating	10% to 90% with a maximum wet bulb temperature of 28°C (82°F) and a minimum dew point of 2°C (36°F) with no condensation
Storage/shipping	5% to 95% with no condensation
Altitude	
Operating	Sea level to 2.4 km (8,000 ft) Maximum operating temperatures decrease by a factor of 1.8°C/1000 (1°F/1000 ft) for operation above sea level.
Storage/shipping	Up to 9.1 km (30,000 ft) above sea level (actual or effective by means of cabin pressurization)
Shock	5 Gs peak at 7 to 13 ms duration in three axes mutually perpendicular (maximum)

ELECTRICAL SPECIFICATIONS

Power consumption

+5.0 V at 8 A nominal
−5.2 V at 2 A nominal
−2.0 V at 1 A nominal

VAXBI BUS SPECIFICATIONS

Bus Characteristics

Type	Synchronous
Width	32 data bits
Cycle time	200 ns
Priority arbitration	Distributed embedded
Parity	Odd
Data transfers	Block mode (masked) Longword Quadword Octaword

Transmission Characteristics

Bandwidth	
Master port	11.4 Mbytes/s
Slave port	13.3 Mbytes/s
Length (maximum)	1.5 m (5 ft)
Bus loading (maximum)	16 logical nodes

CI BUS SPECIFICATIONS

Bus Characteristics

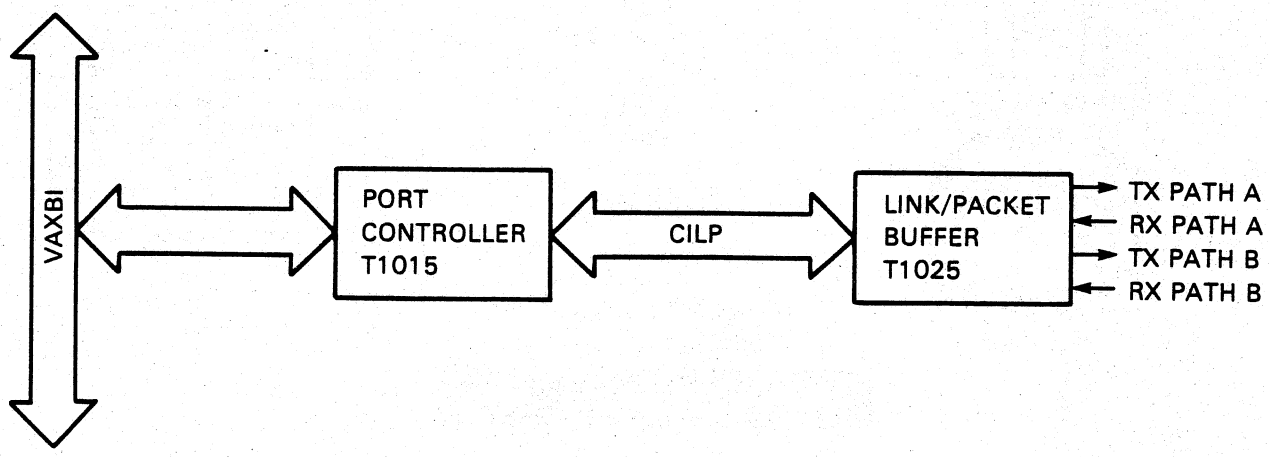
Width	Serial
External length (maximum)	45 m (147.64 ft) radius from star coupler
Data transfer rate	70 Mbits/s (maximum)
Bus loading (maximum)	16 logical nodes
Cable type (BNCIA-XX)	Double shielded coaxial
Cable impedance	50 ohms

1.4 PHYSICAL HARDWARE DESCRIPTION

Refer to Table 1-1 and Figure 1-2 for an overview of the hardware components of the CIBCA.

Table 1-1 Hardware Components of the CIBCA Adapter

Part Number	Component Type
T1015	Port controller module
T1025	Link/packet buffer module
17-01504-01	2.0 inch cable (1 each)
17-01504-02	0.7 inch cable (1 each)
17-01473-01	Internal CI bulkhead cable assembly
12-27249-01	Dummy connector (Receive)
12-27249-02	Dummy connector (Transmit)
12-22246-01	Transition connector assembly (2 each)



MKV87-1039

Figure 1-2 Simplified CIBCA Block Diagram

CIBCA HARDWARE

Refer to Figure 1-3. The CIBCA consists of two T-series type modules. The T-series modules are housed in two adjacent slots within an H9400-A VAXBI card cage. These modules are used to interface the host system's VAXBI bus to the CI bus.

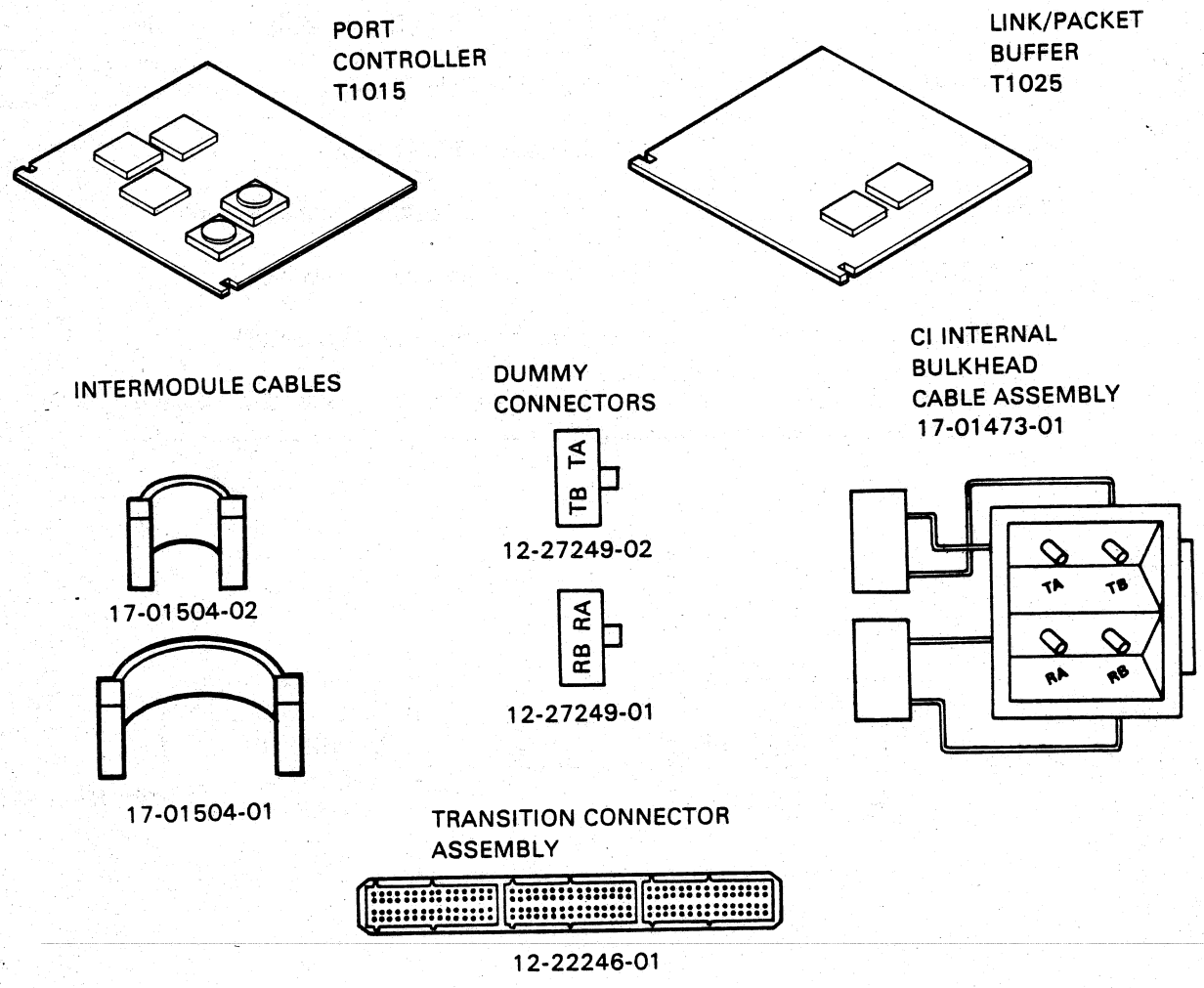


Figure 1-3 Hardware Components of the CIBCA

The Port Controller Module
The port controller module, part number T1015, contains the VAXBI protocol, the VAXBI control logic, microprocessor, and control store.

The Link/Packet Buffer Module
The link/packet buffer module, part number T1025, contains the CI bus protocol logic and CI packet buffer memory.

CIBCA Cables
The two CIBCA cables are used to electrically interconnect the T1015 and T1025 modules. Each cable consists of two 30-pin female connectors and an interconnecting ribbon cable arranged in a ground-signal-ground-fashion. The cables are mated to cable connectors located on the VAXBI card cage corresponding to Zone C of the modules. The short cable completes the innermost electrical connection while the longer cable completes the outermost electrical connection between the two modules.

The CI Bulkhead Connector Panel
The CIBCA is connected internally from the VAXBI backplane to the CI bulkhead connector panel via two pairs of coaxial cables. The CI bulkhead connector panel provides the electrical isolation for the system by creating an EMI/RFI shield without compromising signal integrity. The panel is mounted in the cable connector openings located on the rear inside I/O panels of the cabinet. Two pairs of double-shielded coaxial cables connect the CI paths of the node from the CI bulkhead connector panel to the star coupler. One cable of each pair is for transmitting data, the other for receiving data.

1.5 REFERENCE DOCUMENTS

Title	Document Number
CIBCA Maintenance Print Set	MP-01836-01
CIBCA Hardware Technical Description	EK-CIBCA-TD
SC008 Star Coupler User's Guide	EK-SC008-UG

DIGITAL personnel may order hardware documents from:

Digital Equipment Corporation
10 Forbes Road
Northboro, MA 01532-2597

Attn: Printing and Circulation Services (NRO3/W3)
Order Processing Section

Customers may order hardware documents from:

Digital Equipment Corporation
Peripherals and Supplies Group
Cotton Road
Nashua, NH 03063-1260

CHAPTER 2
SITE PREPARATION AND INSTALLATION

2.1 INTRODUCTION

This chapter contains information on site preparation and installation, including:

Operating Environment – Verifying that the CIBCA option meets all of the minimum physical, environmental, and grounding specifications.

System Configurations – Configuring the CIBCA option to various VAX systems with a VAXBI installed.

Unpacking and Inventory – Unpacking and verifying that the shipment is complete and undamaged.

Installation and Configuration – Installing the modules, intermodule cables, and internal CI bulkhead cable assembly; and configuring the node address of the CIBCA adapter option and other jumper selectable parameters.

2.2 OPERATING ENVIRONMENT

2.2.1 Physical Elements

The CIBCA option requires two adjacent VAXBI slots.

There must be two available 5.1 cm × 10.2 cm (2 in × 4 in) or one 10.2 cm × 10.2 cm (4 in × 4 in) I/O connector panel opening(s) in the cabinet for the CI bulkhead cable connector panel.

2.2.2 System Environment and Grounding Elements

Consult the applicable system installation manual for information regarding system environment and grounding requirements.

2.3 SYSTEM CONFIGURATIONS

Refer to Figures 2-1, 2-2, and 2-3.

NOTE

Ensure that the CIBCA hardware and microcode revision level is consistent with the revision level of the cluster and vice versa. Consult the VAXcluster System Revision Document for more information.

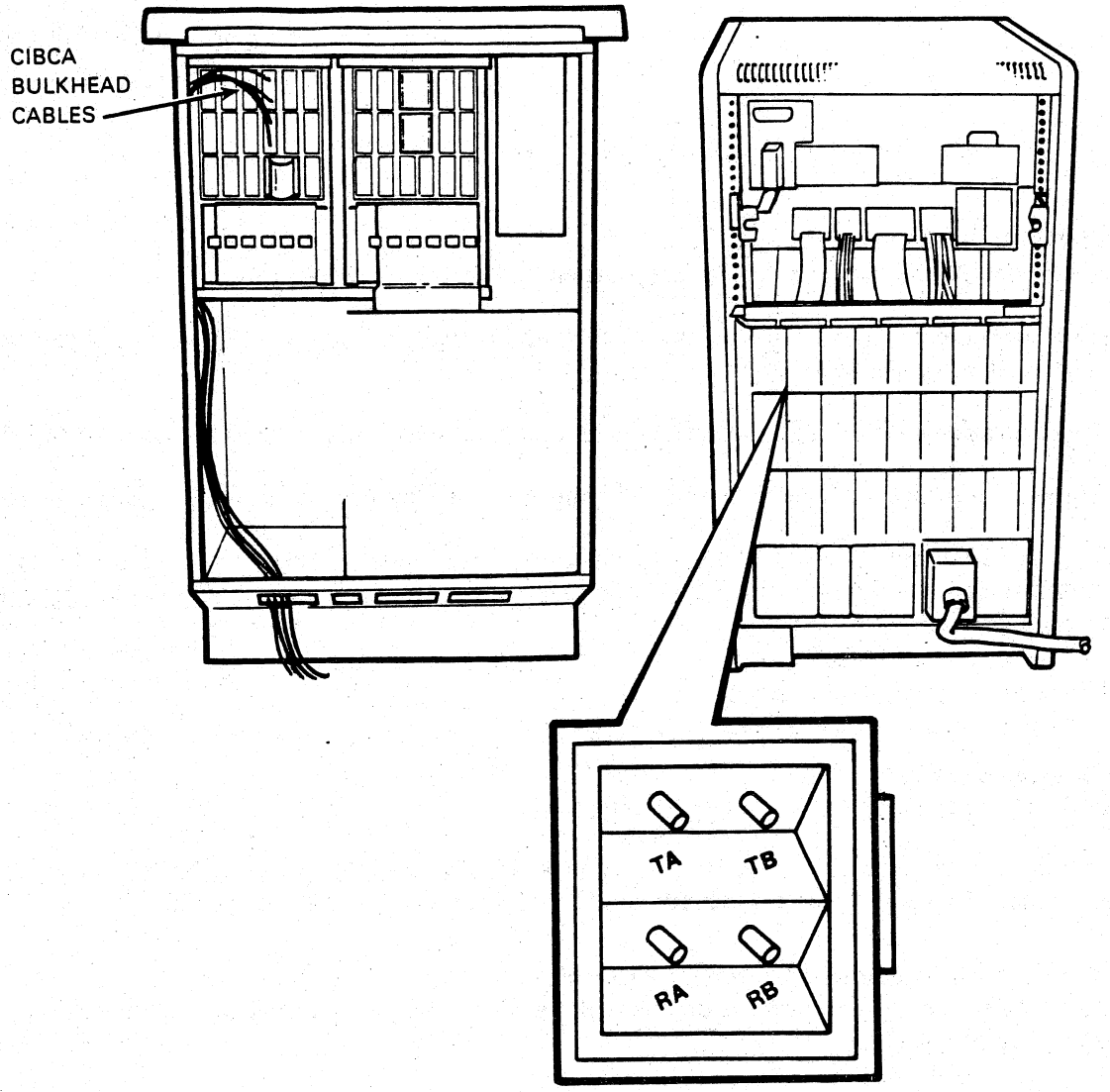


Figure 2-1 CIBCA Adapter – VAX 8200/8300

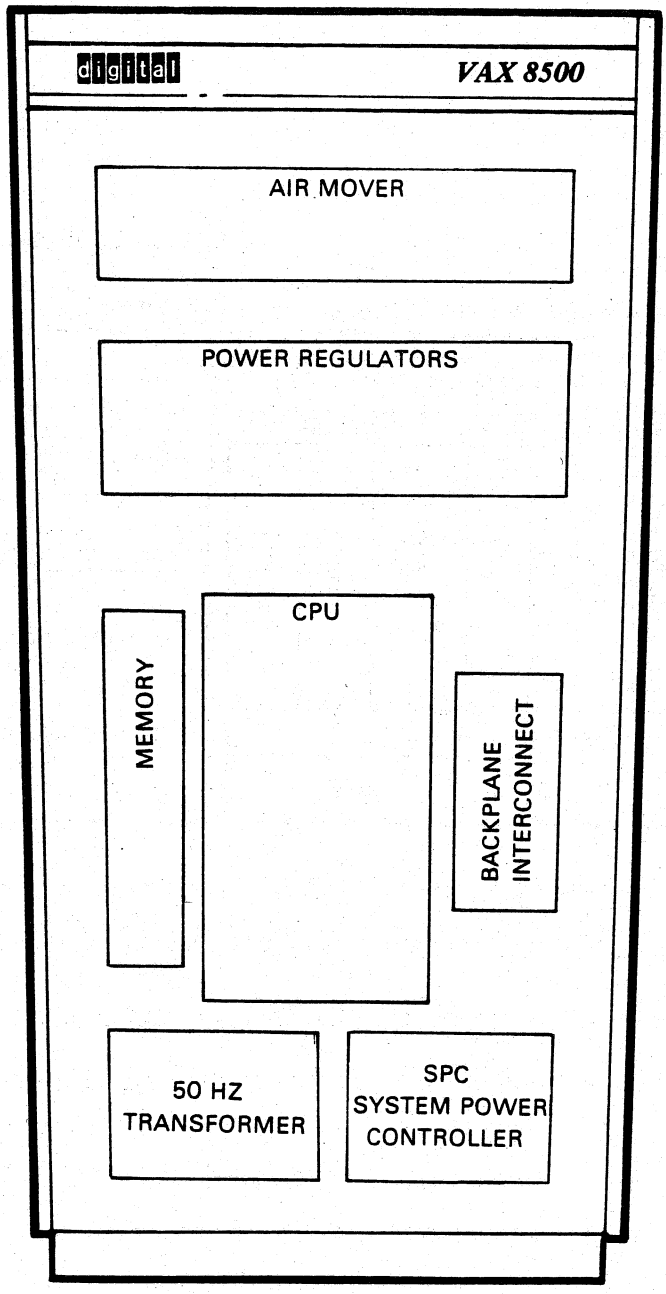


Figure 2-2 CIBCA Adapter - VAX 8500

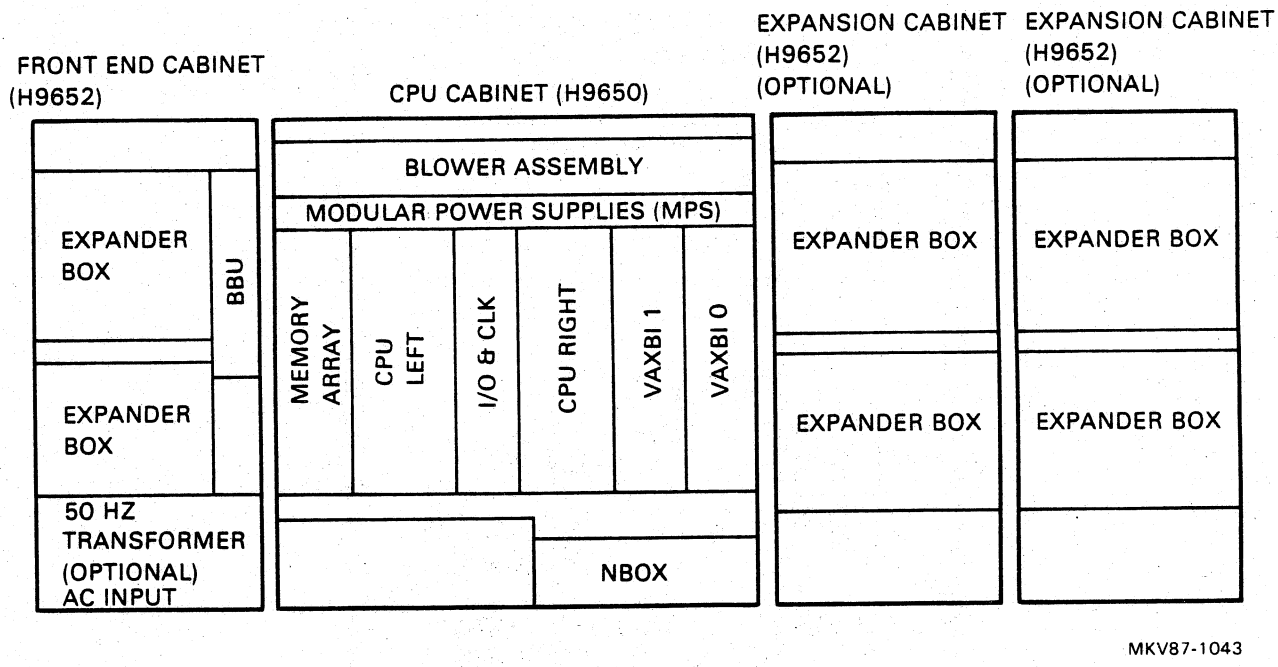


Figure 2-3 CIBCA Adapter – VAX 8800

2.4 UNPACKING AND INVENTORY

The customer is responsible for the actual movement of the equipment to the installation site.

For all VAXBI systems, ensure that all equipment for the CIBCA option is moved to the designated installation site.

Verifying Shipment Inventory

PROCEDURE

1. Inventory all equipment against the shipping list accompanying the equipment.
2. Notify the customer of any opened cartons or boxes and document this fact on the installation report.
3. Notify the Field Service unit manager of any missing or incorrect items.
4. Request that the customer contact the shipping carrier to locate any missing items.
5. Request that the Field Service unit manager check with the Digital Equipment Corporation Traffic and Shipping Department if the shipping carrier does not have the missing items.
6. Check all boxes for external damage (dents, holes, or crushed corners).
7. Notify the customer of all damage and list all damage on the installation report.

Unpacking the Shipping Boxes

PROCEDURE

1. Locate and open the box marked "OPEN ME FIRST". It contains the shipping/accessory list.
2. Open all remaining boxes and inventory the contents against the shipping/accessory list.
3. Inspect the equipment for damage. Report any damage and note it on the installation report.
4. If damage is extensive, notify Digital Equipment Corporation for instructions on how to proceed.

2.5 CIBCA INSTALLATION AND CONFIGURATION

2.5.1 T1015 and T1025 Module Installation

CAUTION

Use an antistatic wrist ground strap (Velostat™ Kit, P/N 29-11762-00) while working on a VAXBI system with its covers removed or when handling any VAXBI module. Do not remove any module from its antistatic packaging until you are ready to install it.

Before installation of the CIBCA, perform the following steps.

1. For CIBCA add-ons, the CIBCA.BIN microcode must be resident on the system console boot device. Before shutting the system down, copy CIBCA.BIN from the appropriate media (supplied with the CIBCA) to the console boot device. Contents of the supplied media are listed below.

   VAX CIBCA MICROCODE UPDATE FLOPPY

   This floppy contains the latest version of the microcode file, CIBCA.BIN. It has the EEPROM Programming and Update utility (EVGDA) which is used to update the CIBCA EEPROM. This floppy is also used to place and/or update the CIBCA microcode on the system console boot device.

Floppy Directory	Description
EVGDA.EXE	CIBCA EEPROM PROGRAM UTILITY
EVGDA.HLP	HELP FILE FOR EVGDA
CIBCA.BIN	CIBCA MICROCODE FILE
VMB.EXE	VAX PRIMARY BOOT FILE (with CIBCA patch until VMS V4.6 is released)

2. Turn OFF power to the system.

3. Ground yourself.

4. Expose the VAXBI card cage.

NOTE

Proceed directly to Section 2.5.2 if the CI bulkhead cable assembly is installed and the VAXBI card cage contains the T1015 module, T1025 module, and the CIBCA intermodule cables.

Velostat is a trademark of the Minnesota Mining and Manufacturing Co.

PROCEDURE

1. Carefully insert the T1015 and T1025 modules into any two unoccupied but adjacent module slots within the same VAXBI card cage (see Figure 2-4). The T1015 is inserted in the lower numbered slot. If the present system cable configuration interferes with the installation of the CIBCA, cables, and jumpers, then the T1015 and T1025 module slot positions (including jumpers and cables) can be interchanged, however, this is *not* recommended.

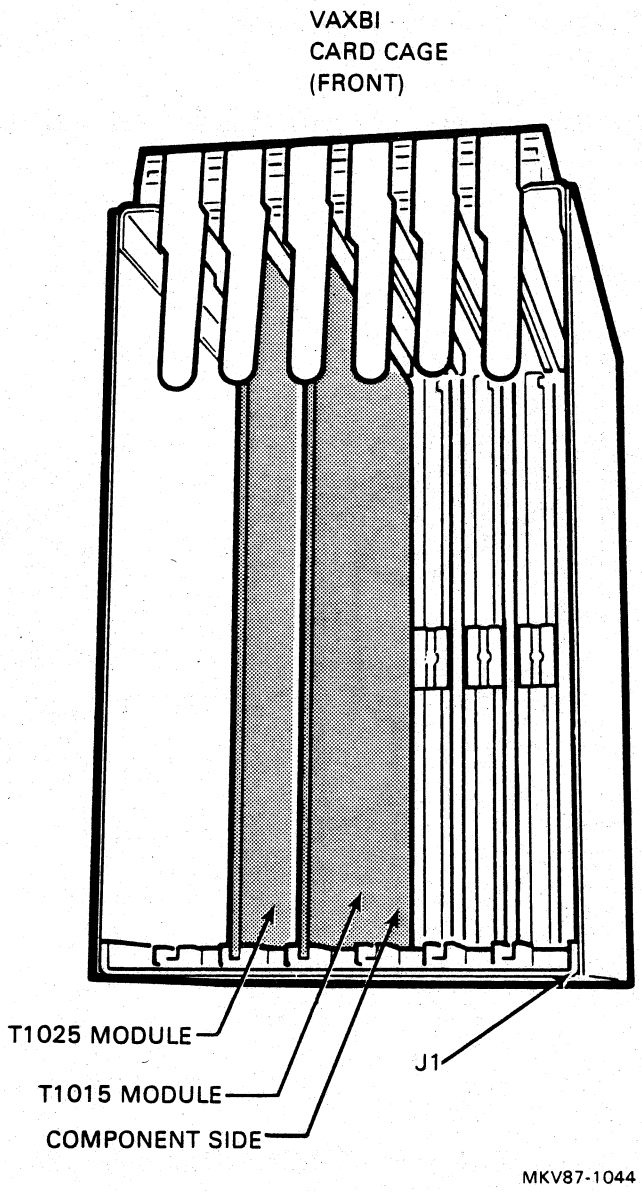


Figure 2-4 VAXBI Card Cage – Module Installation

2. If necessary, install the two transition connector assemblies, DEC P/N 12-22246-01, onto the VAXBI backplane on the slots containing the T1015 and T1025 modules. Using the torque screwdriver supplied in the VAXBI tool kit, DEC P/N 29-25608-00, torque the transition connector assemblies to 5 in/lbs ±1 in/lbs.

3. Refer to Figure 2-5 while carefully connecting the CIBCA intermodule cables to the VAXBI card cage connectors as follows:

   Attach a 0.7 inch cable at Zone C between the innermost connectors of the T1015 and T1025 modules.

   Form the 2.0 inch cable as shown and attach at Zone C between the outermost connectors of the T1015 and T1025 modules.

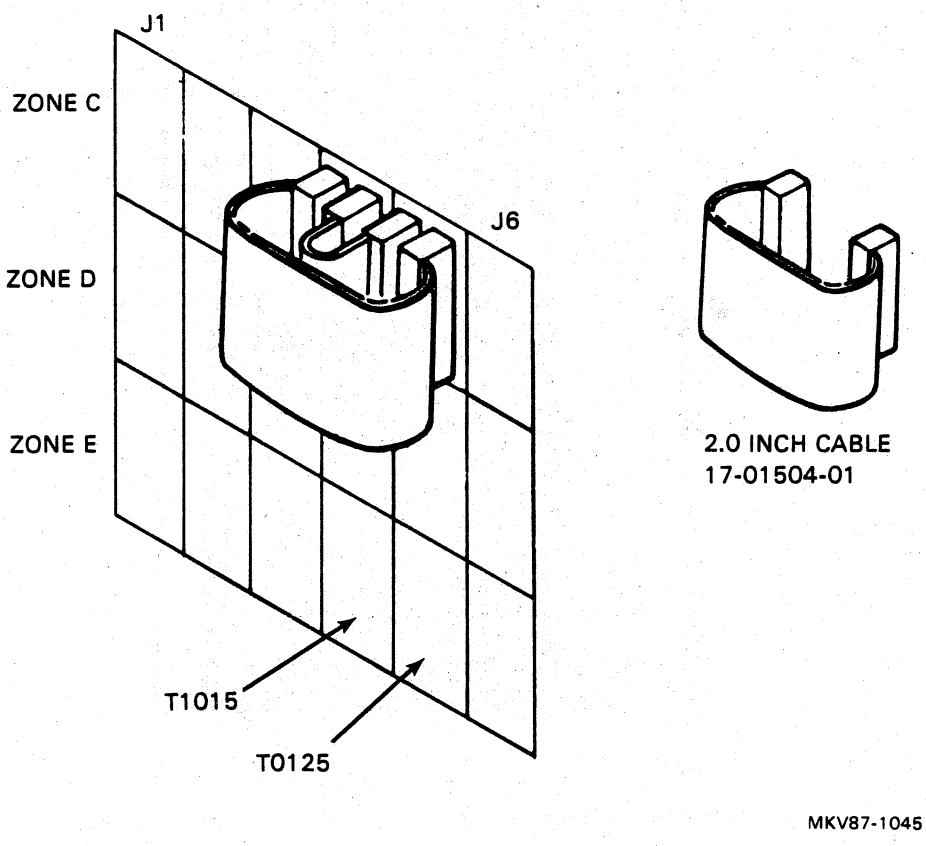


Figure 2-5 . VAXBI Backplane – CIBCA Intermodule Cable Connections

4. Refer to Figure 2-6 while carefully connecting the internal CI bulkhead cables.

5. Route the internal CI bulkhead cables and install the I/O bulkhead panel.

6. Install the RECEIVE dummy connector in Zone E of the slot containing the T1025 (solder side). Install the TRANSMIT dummy connector in Zone D of the slot containing the T1025 (solder side).

7. Install the RECEIVE cable connector into the transition header in Zone E of the slot containing the T1025 (component side). Install the TRANSMIT cable connector into the transition header in Zone D of the slot containing the T1025 (component side). Cables must exit toward Zone E. Use tie-wraps to secure cables.

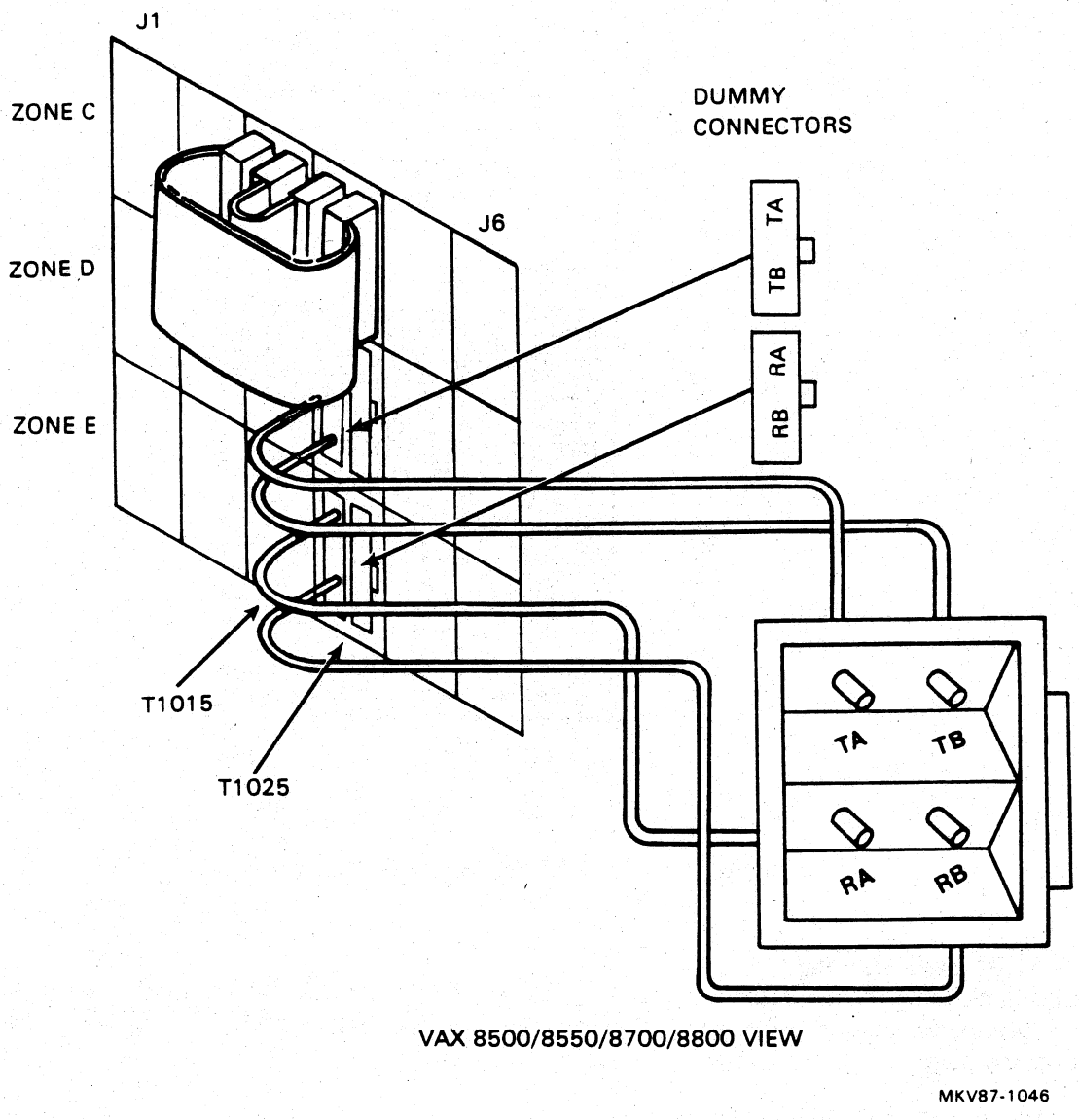


Figure 2-6 CIBCA CI Internal Bulkhead Cable Assembly Installation

2.5.2 VAXBI Backplane Jumper Verification

Several jumpers and a VAXBI node ID encapsulated plug are used on the user section of the VAXBI backplane to control certain operating parameters for the CIBCA. These jumpers are configured at the factory for normal operation when the CIBCA is shipped installed in the system. The configuration settings of the CIBCA backplane jumpers must match the parameters selected in the other VAXcluster nodes. Figure 2-7 illustrates a VAXBI backplane assembly.

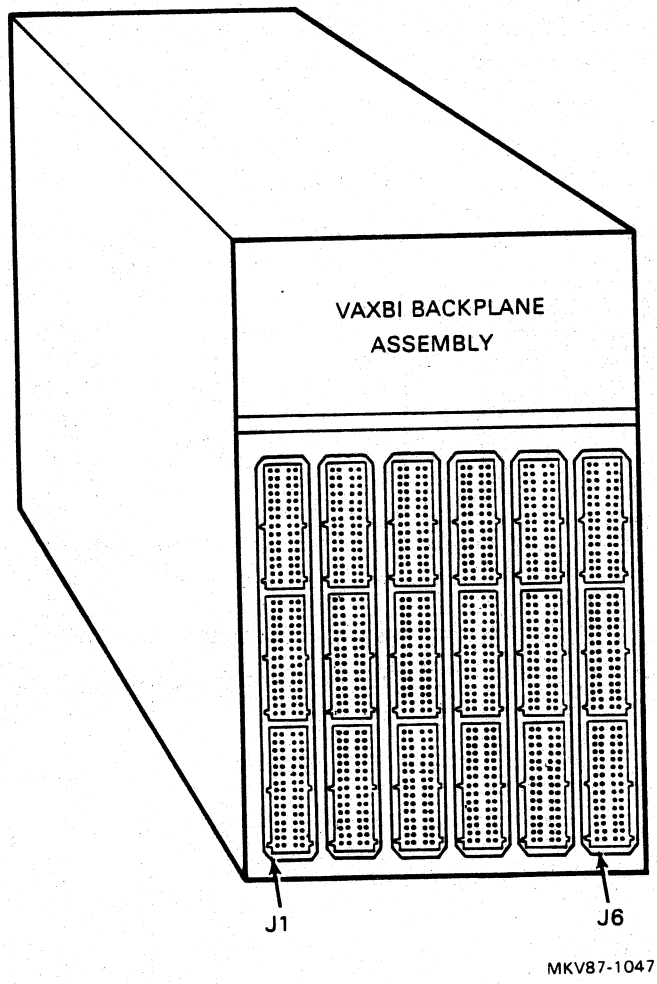


Figure 2-7 VAXBI Stationary Backplane

2.5.3 VAXBI Node ID Plug

An encapsulated plug on the VAXBI backplane (on the T1015 module slot) provides an encoded 4-bit binary identification number. The decimal equivalent identification number or VAXBI NODE ID is printed on the plug.

2.5.4 CI Node Address Jumpers

The CI node address is obtained from the T1015 backplane slot. The CI node address and its complement must be configured exactly the same. Table 2-1 lists the way the CI node address jumpers should be configured for a respective address. See Figure 2-8 for a detailed view of CIBCA jumper backplane pinning.

Table 2-1 CI Node Address Jumpers

D5-D35 E1-E31 <2^7>	D6-D36 E2-E32 <2^6>	D7-D37 E3-E33 <2^5>	D8-D38 E4-E44 <2^4>	D9-D39 E5-E35 <2^3>	D10-D40 E6-E36 <2^2>	D11-D41 E7-E37 <2^1>	D12-D42 E8-E38 <2^0>	CI NODE ADDRESS DECIMAL
OUT	OUT	OUT	OUT	OUT	OUT	OUT	OUT	0
OUT	OUT	OUT	OUT	OUT	OUT	OUT	IN	1
OUT	OUT	OUT	OUT	OUT	OUT	IN	OUT	2
				.				
				.				
				.				
IN	IN	OUT	IN	IN	IN	IN	OUT	222
IN	IN	OUT	IN	IN	IN	IN	IN	223
IN	IN	IN	OUT	OUT	OUT	OUT	OUT	224

NOTE:
Addresses above 223 decimal are illegal.

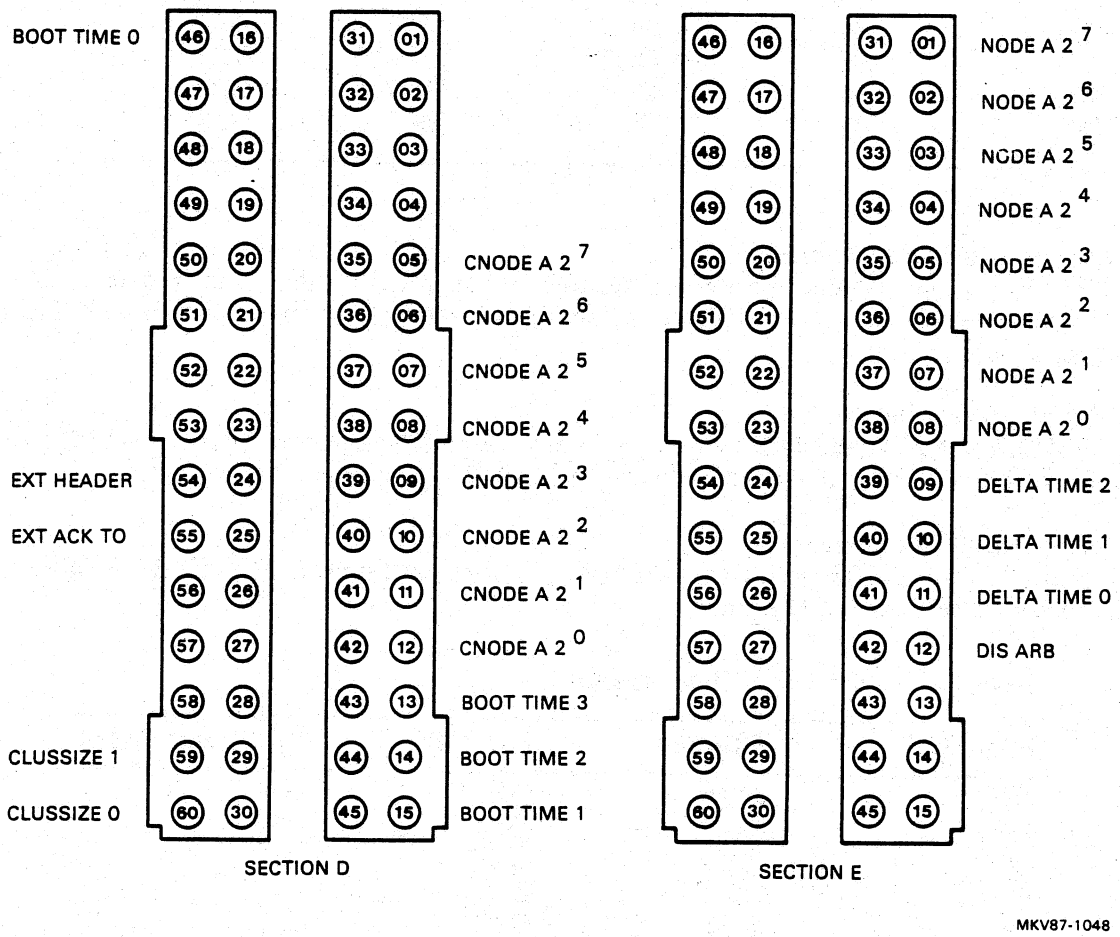


Figure 2-8 CIBCA Backplane Jumper Pinning

2.5.5 Boot Time Jumpers

Boot time is the length of time the CIBCA adapter waits after power up before exiting the CI UNINIT state. Table 2-2 identifies jumper positions to select the desired boot time.

Table 2-2 Boot Time Jumpers

D13-D43	D14-D44	D15-D45	D16-D46	Time in Seconds
OUT	OUT	OUT	OUT	1500 (Default)
OUT	OUT	OUT	IN	1400
OUT	OUT	IN	IN	1300
OUT	OUT	IN	OUT	1200
OUT	IN	OUT	OUT	1100
OUT	IN	OUT	IN	1000
OUT	IN	IN	OUT	0900
OUT	IN	IN	IN	0800
IN	OUT	OUT	OUT	0700
IN	OUT	OUT	IN	0600
IN	OUT	IN	OUT	0500
IN	OUT	IN	IN	0400
IN	IN	OUT	OUT	0300
IN	IN	OUT	IN	0200
IN	IN	IN	OUT	0100
IN	IN	IN	IN	0000

2.5.6 Disable Arbitration Jumpers

Jumper E12-E42 is the Disable Arbitration jumper. When inserted, this jumper defeats the normal arbitration sequence and allows the T1025 to transmit after waiting only one basic quiet slot (Delta) time. This jumper is normally out.

2.5.7 Extend Header Jumper

Jumper D24-D54 is the Extend Header jumper. When inserted, this jumper allows the T1025 to extend the number of bit synchronous characters in the header. This jumper is normally out.

2.5.8 Alter Delta Time Jumpers

These jumpers force the T1025 to increase the basic CI quiet slot Delta time. Table 2-3 identifies jumper insertion to select the desired Delta time.

Table 2-3 Alter Delta Time Jumpers

E09-E39	E10-E40	E11-E41	Quiet Slot Count
OUT	OUT	OUT	7
OUT	OUT	IN	10
OUT	IN	OUT	14
OUT	IN	IN	16
IN	OUT	OUT	21
IN	OUT	IN	25
IN	IN	OUT	32
IN	IN	IN	Illegal

2.5.9 Node Count Jumpers

These jumpers force the CI arbitration logic to arbitrate for more than 16 nodes. Table 2-4 identifies the jumper insertions to select the node count.

Table 2-4 Node Count Jumpers

D29-D59	D30-D60	Node Count (In Decimal)
OUT	OUT	16
OUT	IN	32
IN	OUT	64
IN	IN	128

2.5.10 Extend ACK Timeout Jumper

Jumper D25-D55 is the Extend ACK Timeout jumper. When installed, this jumper forces the T1025 to increase the timeout period for a CI ACK return. This jumper is normally out.

CHAPTER 3
ACCEPTANCE VERIFICATION

3.1 INTRODUCTION

Diagnostic Verification – Verifying the CIBCA hardware operation by running diagnostic tests with the system in a standalone environment.

Maintenance Verification – Facilitating VAXcluster maintenance by describing the tools that are required and/or provided for individual nodes or options within a VAXcluster system.

3.2 POWER-ON PROCEDURE

Refer to the appropriate system user guide for power-on procedures.

3.3 DIAGNOSTIC VERIFICATION

To determine if the CIBCA adapter hardware is functioning properly, seven Level 3 diagnostic programs must be executed. These diagnostic programs and their applicable diagnostic supervisor program, are contained on separate RX50 floppy diskettes.

Five of the seven Level 3 diagnostic programs are executed with the system operating in a standalone environment (*not* connected to a VAXcluster and *not* running under the VMS operating system).

Two of the seven Level 3 diagnostic programs are then executed with the system operating in a standalone environment (*not* running under the VMS operating system) but connected to the coupler in order to verify the functional CIBCA hardware operation within the system. This is referred to as functional level testing.

Table 3-1 lists the CIBCA diagnostic programs. Table 3-2 provides a summary of the CIBCA diagnostic testing hierarchy. Figure 3-1 describes the CIBCA acceptance testing flow.

Table 3-1 List of CIBCA Diagnostic Programs

Program Designation	Program Title
EVGCA	CIBCA T1015 Repair Level Diagnostic 1
EVGCB	CIBCA T1015 Repair Level Diagnostic 2
EVGCC	CIBCA T1015 Repair Level Diagnostic 3
EVGCD	CIBCA T1015 Repair Level Diagnostic 4
EVGCE	CIBCA T1025 Repair Level Diagnostic 1
EVGCK	CIBCA Repair Level Microcode 1
EVGCL	CIBCA Repair Level Microcode 2
EVGCM	CIBCA Repair Level Microcode 3
EVGCN	CIBCA Repair Level Microcode 4
EVGDA	CIBCA EEPROM Programming and Update Utility
EVGAA	CI Functional Diagnostic 1
EVGAB	CI Functional Diagnostic 2
EVXCI	CI Exerciser Diagnostic

Table 3-2 CIBCA Diagnostic Testing Hierarchy

Diagnostic Category	Diagnostic Program Level	Testing Function
Repair	3	Tests the detailed hardware operation of the CIBCA adapter.
Functional	3	Tests the functional hardware operation of the CIBCA adapter.
Exerciser	2R	Tests the communications between CI nodes.
		Detects a failing CI node.
		Verifies repair of a failing CI node.

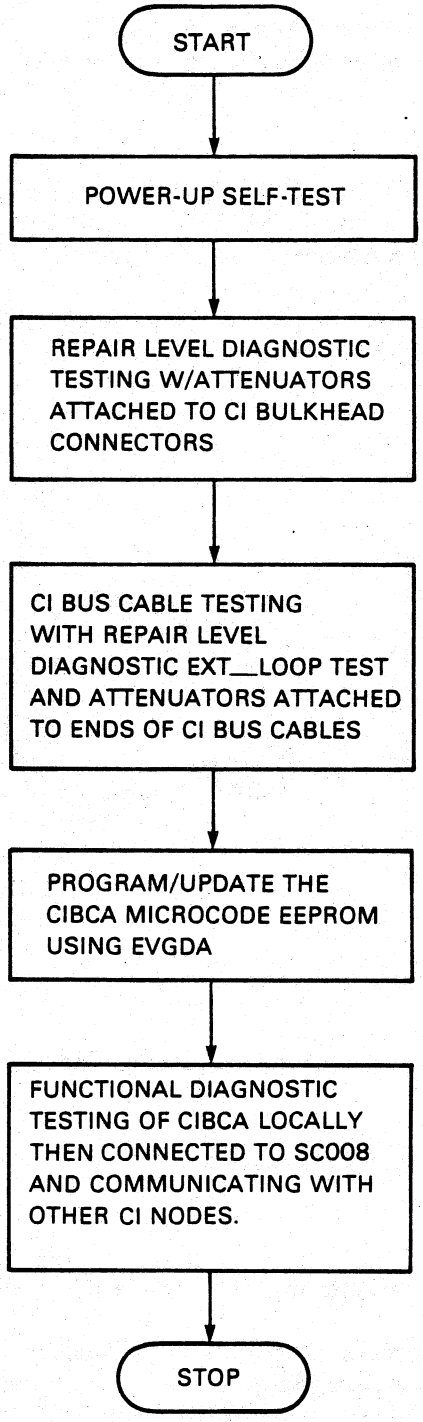


Figure 3-1 CIBCA Acceptance Testing Flow Diagram

3.3.1 Preliminary Setup

Before running the diagnostics, make the following CI bus loopback connections on the CI bulkhead connector panel located at the back of the cabinet (see Figure 3-2).

PROCEDURE

1. Using one of the attenuator pads (P/N 12-19907-01) and two of the modularity cables (P/N 70-18530-00) supplied in the CIBCA Controlled Distribution (CD) kit, DEC P/N A2-W1207-10, connect TRANSMIT A to RECEIVE A.

2. Perform the same connection for path B using the other attenuator pad and two modularity cables from the CIBCA CD kit. Connect TRANSMIT B to RECEIVE B.

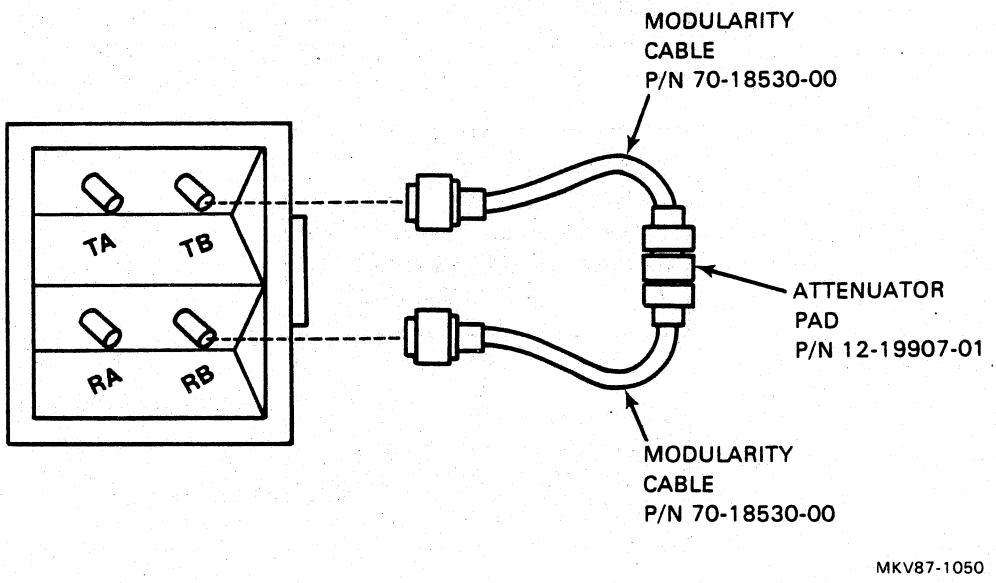


Figure 3-2 Diagnostic Loopback Cable Connections

3.3.2 Loading the Diagnostic Supervisor Program

PROCEDURE

1. Insert the proper RX50 diskette into the console RX50 disk drive unit 0.

2. Load the diagnostic supervisor program into physical memory from the console load device. This procedure may vary depending on the system type in which the CIBCA is installed. Consult the applicable system installation manual for diagnostic supervisor load and run procedures.

   The following are examples to load the diagnostic supervisor.

   VAX 8800 SYSTEMS

   >>>@DIABOO (boots the diagnostic supervisor from the VAX console fixed disk)

VAX 8200 SYSTEMS

Insert the diagnostic supervisor diskette into the left RX50 drive, then

>>> B CSA1 (boots the diagnostic supervisor from the console load device).

3. Identify the CIBCA adapter and its node configuration parameters to the diagnostic supervisor program. The ATTACH/SELECT sequence will once again vary depending upon the system type in which the CIBCA is installed. It is assumed that the installer is familiar with the diagnostic supervisor ATTACH and SELECT sequences for VAXBI devices on the processor being used. Below are examples for attaching the CIBCA hardware in VAX 8800 and VAX 8200 series system environments.

CIBCA ON A VAX 8800

a. All ATTACHes specifying BI NODE numbers should be documented to show that node numbers are to be in hexadecimal.

b. BI nodes instead of being attached to HUB should be attached to NBIBn, that is, the following sequence should be added:

```
ATTACH NBIA HUB NBIAn                ! where n 0 or 1
LOGICAL ADAPTER # ? N                ! 0 or 1

ATTACH NBIB NBIAn
Device Name? NBIBn                   ! where n 0 to 3
BI # ? n                             ! 0 or 1
BI Node Number (HEX) ? n             ! n 0 to F

ATTACH                               ! Then attach UUT
Device name? CIBCA                   ! Device to test
Device link? NBIBn                   !
Device? PAAn                         ! Specify the CI
BI Node (DEC) ? n                    ! BI node #
BR Level? 4                          ! BR level 4
CI address (DEC) ? n                 ! 0 - 16
```

For example:

```
ATTACH NBIA HUB NBIA0 0              ! logical unit 0
ATTACH NBIB NBIA0 NBIB0 1 0          ! BI #1, BI node 0
ATTACH CIBCA NBIB0 PAA0 2 4 0        ! BI node 2, BR level 4
                                     ! CI node 0
SELECT PAA0                          ! device to be run
```

CIBCA ON A VAX 8200

```
ATTACH                               ! Then attach UUT
Device name? CIBCA                   ! Device to test
Device link? HUB                     !
Device? PAAn                         ! Specify the CI
BI Node (DEC) ? n                    ! BI node #
BR Level? n                          ! BR level #
CI address (DEC) ? n                 ! 0 - 16
```

4. Select the CIBCA adapter as the unit under test, as follows:

   DS> SELECT PAA0

5. Show the unit selected, as follows:

   DS> SHOW SELECT

3.3.3 Repair Level Testing

A minimum of five successful passes of each diagnostic program must be completed to satisfy acceptance testing requirements. Examples 3-1 through 3-5 provide trace printouts for diagnostics EVGCA through EVGCE, respectively.

NOTE

HELP files are available under the diagnostic supervisor for all the diagnostic programs including the supervisor program itself.

PROCEDURE

1. While proceeding through the diagnostic acceptance testing, ensure that the diagnostics are accessible via the DEFAULT LOAD PATH. This may require changing diagnostic media in the current load path device.

2. Load the EVGCA diagnostic program, as follows:

   DS> LOAD EVGCA (first repair level diagnostic)

3. Set the desired diagnostic supervisor control flags to enable printing of the number and title of each test before it is executed and to halt on a detected error.

   DS> SET FLAGS TRACE, HALT

4. Start the diagnostic program.

   DS> START/PASS:5

5. Repeat Steps 1 through 4 to load and execute the remaining repair level diagnostics (EVGCB through EVGCE). Substitute the target filename in Step 2.

```
DS> ATTACH CIBCA HUB PAA0 12 4 0
DS> SEL PAA0
DS> SET TRACE, HALT
DS> LOAD EVGCA DS> START

.. Program: EVGCA - T1015 CIBCA Repair Level Diagnostic Part
1, Revision 1.0, 13 tests, at 11:45:34.75. Testing: _PAA0

Test 1:  Device Type/BIIC Configuration Register Test
Test 2:  BI Control and Status Register Test
Test 3:  BCA BI Required Register Test
Test 4:  BCA General Purpose Device Register Test
Test 5:  BCA User CSR Space Register Test
Test 6:  Port Status Register Test
Test 7:  BCA Specific Register Test
Test 8:  Local Store/VCDT Address Read/Write Test
Test 9:  Local Store/VCDT Data Read/Write Test
Test 10: Local Store/VCDT Dynamic Memory Test
Test 11: Control Store Address Test
Test 12: Control Store Read/Write Ram Test
Test 13: Control Store Ram Dynamic Memory Test
.. End of run, 0 errors detected, pass count is 1, time is
24-OCT-1986 11:47:09.98
```

Example 3-1 Trace Printout for Repair Level Diagnostic EVGCA

```
DS> LOAD EVGCB
DS> START

.. Program: EVGCB - T1015 CIBCA Repair Level Diagnostic Part
2, Revision 1.0, 17 tests, at 11:49:36.98. Testing: _PAA0

Test 1:  EEPROM Integrity Verification Test
Test 2:  Internal Bus Branch/Sequencer Jump Test
Test 3:  Microprogram Controller UPC+1 Test
Test 4:  Microprogram Controller JSB/RSB Test
Test 5:  Microprogram Controller Pop Micro Stack Test
Test 6:  Single Operand Instruction Test
Test 7:  Two Operand Instruction Test
Test 8:  Single Bit Shift Instruction Test
Test 9:  Rotate By n Bit Instruction Test
Test 10: Set RAM Bit n Instruction Test
Test 11: Set Ram MINUM Bit n Instruction Test
Test 12: Set ACC Minus Bit n Instruction Test
Test 13: Set DLATCH Bit n Instruction Test
Test 14: Reset RAM Bit n Instruction Test
Test 15: Reset ACC Bit n Instruction Test
Test 16: Reset DLATCH Bit n Instruction Test
Test 17: Test RAM Bit n Instruction Test
.. End of run, 0 errors detected, pass count is 1, time is
24-OCT-1986 11:50:00.02
```

Example 3-2 Trace Printout for Repair Level Diagnostic EVGCB

```
DS> LOAD EVGCC
DS> START

.. Program: EVGCC - T1015 CIBCA Repair Level Diagnostic Part
3, Revision 1.0, 18 tests, at 11:50:13.17. Testing: _PAA0

Test 1:  Test ACC Bit n Instruction Test
Test 2:  Test DLATCH Bit n Instruction Test
Test 3:  Load RAM Bit n Instruction Test
Test 4:  Load RAM NOT Bit n Instruction Test
Test 5:  Load ACC Bit n Instruction Test
Test 6:  Load ACC NOT Bit n Instruction Test
Test 7:  Load YBUS Bit n Instruction Test
Test 8:  Load YBUS NOT Bit n Instruction Test
Test 9:  Add RAM Bit n Instruction Test
Test 10: Add ACC Bit n Instruction Test
Test 11: Add DLATCH Bit n Instruction Test
Test 12: Rotate and Merge by n Instruction Test
Test 13: Rotate and Compare by n Instruction Test
Test 14: Prioritize Instruction Test
Test 15: CRC Instruction Test
Test 16: No Operation Instruction Test
Test 17: AMD29116 Internal Register Address Test
Test 18: Local Store/VCDT Microcode Access Test
.. End of run, 0 errors detected, pass count is 1, time is
24-OCT-1986 11:50:59.24
```

Example 3-3 Trace Printout for Repair Level Diagnostic EVGCC

```
DS> LOAD EVGCD
DS> START

.. Program: EVGCD - T1015 CIBCA Repair Level Diagnostic Part
4, Revision 1.0, 23 tests, at 11:45:34.75. Testing: _PAA0

Test 1: Register Dual Address Test
Test 2: Local Store/Virtual Circuit Descriptor Table Parity
Error Test
Test 3: Interrupt Test
Test 4: MTE During Interrupt Test
Test 5: Microword Verification Test
Test 6: Control Store Parity Error (CSPE) Test
Test 7: Condition Code Branch MUX Test
Test 8: Maintenance Timer Disable Branch Test
Test 9: Tick Branch Test
Test 10: IB Register Read/Write Loopback Test
Test 11: BCAI Register Test
Test 12: Register Written Test
Test 13: BI Master Write/Read Test
Test 14: Power Fail Test with Power Fail Disable Set Test
Test 15: CBOR/CBIR Port Initiated Loopback Test
Test 16: Command Address/Byte Count Register
Test 17: Datamover Loopback Test
Test 18: Datamover Read/Write to Host Memory Test
Test 19: Page Overflow Test
Test 20: Write/Stop Command Test
Test 21: BI Slave Transaction Test
Test 22: Suspend and Release II Bus Test
Test 23: Suspend and Release CILP Bus Test
.. End of run, 0 errors detected, pass count is 1, time is
24-OCT-1986 11:50:25.08
```

Example 3-4 Trace Printout for Repair Level Diagnostic EVGCD

```
DS> LOAD EVGCE
DS> START

.. Program: EVGCE - CIBCA T1025 Repair Level Diagnostic
 Revision 1.0, 15 tests, at 11:51:52.34. Testing: _PAA0

Test 1: Link Configuration & CILP Bus Integrity Test

Contents of CONFIGURATION Register 0

True Node Address : 0E
Cluster Size      : 00
Extended ACK      : 00
Extended Header   : 00
Disable ARB       : 00
Delta Time        : 01

Contents of CONFIGURATION Register 1

Complement Node Address : F1
Boot Time               : 0F

Test 2: Packet Buffer Data Integrity Test
Test 3: Transmit with External/Internal Loopback Set Test
Test 4: Transmit with Internal Loopback Set Test
Test 5: Transmit with External Loopback Set Test
Test 6: Force Transmit Parity Error in Internal Loopback Test
Test 7: Invalid Complement Destination Node Number Test
Test 8: True/Complement Destination Node Number Swap Test
Test 9: Bad CRC Test
Test 10: Negative (NAK) Acknowledgement Test
Test 11: Transmit Abort Test
Test 12: Extended Link Configuration Test
Test 13: Valid CI Node Number Test, All Combinations
Test 14: Internal Interaction Test
Test 15: Arbitration Time Test
.. End of run, 0 errors detected, pass count is 1, time is
24-OCT-1986 11:57:21.16
```

Example 3-5 Trace Printout for Repair Level Diagnostic EVGCE

3.3.4 CI Bus Cable Testing

After successfully completing five passes of each of the five repair level diagnostics, remove the attenuator pads and modularity cables from the CI bulkhead connectors (J21-J24) and perform the following steps.

PROCEDURE

1. Verify that this CIBCA port has a unique node address within the VAXcluster before connecting any cables.

2. Locate the set of four CI bus coaxial cables (BNCIA-XX). Label each end of the CI bus coaxial cables with the CI node and CI path information using the labels provided with the BNCIA-XX cables. Connect one end of each cable to the appropriate CI bulkhead connector.

   NOTE

   The coaxial CI bus cables may be connected to or removed from the CI bulkhead connectors without powering down the system. DO NOT unroll or route the CI bus cables at this time.

3. Connect the two attenuator pads to the free ends of the coaxial CI bus cables. Be sure to connect TRANSMIT A to RECEIVE A, and TRANSMIT B to RECEIVE B.

4. Run five passes of the EXTERNAL_LOOP section of the diagnostic program EVGCE to test the CI bus cables.

   DS> RUN EVGCE/SEC:EXTERNAL_LOOP/PASS:5

3.3.5 CIBCA EEPROM Programming and Update Utility

PROCEDURE

1. Insert the RX50 floppy containing EVGDA and the CIBCA microcode file CIBCA.BIN into the console floppy drive being used as the diagnostic load path.

   By default, tests 3 through 5 are run. Tests 1, 2, 6, and 7 are special purpose tests; these are not normally executed in the field.

2. Load the EVGDA diagnostic program as follows:

   DS> LOAD EVGDA

3. Start the diagnostic program.

   DS> START

   Example 3-6 shows a trace printout when running the (default) update section .

```
.. Program: EVGDA - CIBCA EEPROM Programming and Update
Utility, Revision 1.1, 7 tests, at 10:32:34.78. Testing:
_PAA0

Summary of EEPROM data from File Header
The EEPROM Version Number is: 0001
The EEPROM CRC Value is: 5012E01c
The EEPROM Starting Logical Block Number is: 1
The EEPROM Microcode Size, in Bytes, is: 8190

Summary of Functional data from File Header
The Functional Version number is: 0001
The Functional CRC Value is: 5B9D6A8F
The Functional Starting Logical Block Number is: 17
The Functional Microcode Size, in 16-Bit Words, is: 12288

Test 3: Update the EEPROM Microcode Started Update of the
EEPROM at 29-OCT-1986 10:32:46.05

CURRENT value of EEPROM Update Counter is : 4

NEW value of EEPROM Update Counter is : 5 Finished with
Update of the EEPROM at 29-OCT-1986 10:34:09.39

Test 4: Verify the Contents of EEPROM Started
Verification of the EEPROM at 29-OCT-1986 10:34:10.29
Finished with Verification of the EEPROM at 29-OCT-1986
10:34:10.98

Test 5: Execute and Check Selftest Status Started
Selftest Execution at 29-OCT-1986 10:34:11.99 Finished
Selftest Execution at 29-OCT-1986 10:34:11.99

.. End of run, 0 errors detected, pass count is 1, time
is 29-OCT-1986 10:34:13.24
```

Example 3-6 Trace Printout for Repair Level Diagnostic EVGDA

3.3.6 Functional Level Testing

With the CI bus cables and attenuator pads providing signal loopback, load and run the CI functional diagnostics EVGAA and EVGAB. A minimum of five passes of each diagnostic must be completed to satisfy acceptance testing requirements. Examples 3-7 and 3-8 show trace printouts for diagnostics EVGAA and EVGAB, respectively.

PROCEDURE

1. While proceeding through the diagnostic acceptance testing, ensure that the diagnostics are accessible via the DEFAULT LOAD PATH. This may require changing diagnostic media in the current load path device.

2. Load the EVGAA diagnostic program.

   DS> LOAD EVGAA (first functional diagnostic)

3. Set event flag 1 to load the CIBCA.BIN microcode. This is always required after running the repair level diagnostics.

   DS> SET EVENT FLAG 1

4. Set the desired diagnostic supervisor control flags to enable printing of the number and title of each test before it is executed and to halt on a detected error.

   DS> SET FLAGS TRACE, HALT

5. Start the EVGAA diagnostic program

   DS> START/PASS:5

6. After five successful passes of EVGAA, load and run EVGAB by typing the following.

   DS> LOAD EVGAB
   DS> CLEAR EVENT FLAG 1
   DS> START/PASS:5

7. Disconnect the attenuators from the ends of the CI bus cables in preparation for routing and connecting the cables to the star coupler.

8. Route and connect the CI bus coaxial cables to the coupler.

   NOTE

   For information and connecting of the coaxial CI bus cables to the star coupler, refer to the SC008 Star Coupler User's Guide.

9. Run EVGAA and EVGAB

```
DS> LOAD EVGAA
DS> SET EVENT FLAG 1
DS> SET FLAGS TRACE, HALT
DS> START/PASS:5

.. Program: EVGAA - CI FUNCTIONAL PART I, Revision 4.0,
17 tests, at 11:11:45.99. Testing _PAA0

Event Flag 1 SET = Load CI Microcode
Event Flag 2 SET = Print Queue Entries
Event Flag 3 SET = REQID Loop Function in Test 1

Testing Device _PAA0

EEPROM Revision = 0001  Functional Revision = 0001

Test 1: Cluster Configuration
Contents of the PORT PARAMETER REGISTER is :
PPR:[0FF0010E(X)] ;
CLUSTER_SIZE=16,
IBUF_LEN=0FF0(X),
MBZ=0(X),
DISABLE_ARB=0(X),
EXTENDED_HEADER=0(X),
SLOT_COUNT=10,
PORT_NUMBER=0E(X)

Cluster Configuration for Path A
********************************
You CANNOT Differentiate between a CI780, CI750, or a
CIBCI remotely. (PS = Path Select, TP = Transmit Path, RP
= Receive Path)

Node    Device    Hard  Soft      Port          Path    P T R
Number    Type     Rev.  Rev.  Functionality   Status  S P P
------  ------  ------------  -------------  ------  - - -
02      CIXXX     0007  0007    FFFF0F00(X)     OK    A A A
03      HSC50       0225        4F710200(X)     OK    A A A
0E      CIBCA     0001  0001    FFFF0F00(X)     OK    A A A

Cluster Configuration for Path B
********************************
(PS = Path Select, TP = Transmit Path, RP = Receive Path)

Node    Device    Hard  Soft      Port          Path    P T R
Number    Type     Rev.  Rev.  Functionality   Status  S P P
------  ------  ------------  -------------  ------  - - -
02      CIXXX     0007  0007    FFFF0F00(X)     OK    B B B
03      HSC50       0225        4F710200(X)     OK    B B B
0E      CIBCA     0001  0001    FFFF0F00(X)     OK    B B B
```

Example 3-7 Trace Printout for Functional Diagnostic EVGAA (Sheet 1 of 2)

```
    Nodes NOT Listed do not exist on Cluster

    Test 2: SETCKT with Various Masks and M_Values
    Test 3: SETCKT for Each Valid Port
    Test 4: SETCKT for Invalid Port
    Test 5: REQID Basic
    Test 6: REQID With 6 Packets on DGFQ
    Test 7: Datagram Discard
    Test 8: Response Queue Available Interrupt
    Test 9: Send Datagram
    Test 10: SNDMSG With No Virtual Circuit Open
    Test 11: Send Message Crossing Page Boundary
    Test 12: Message Length Test
    Test 13: Packet Size Violation
    Test 14: Send Loopback (SNDLB)
    Test 15: SNDLB Full Buffer On Path A
    Test 16: SNDLB Full Buffer On Path B
    Test 17: SNDLB Automatic Path Selection
    .. First pass done, 0 errors detected, time is 24-OCT-1986
    11:12:55.07
```

Example 3-7 Trace Printout for Functional Diagnostic EVGAA (Sheet 2 of 2)

```
DS> LOAD EVGAB
DS> CLEAR EVENT FLAG 1, 2
DS> SET FLAGS TRACE, HALT
DS> START/PASS:5

..Program: EVGAB - CI FUNCTIONAL PART II, Revision 4.0, 12 tests
  at 00:00:00.00. Testing: _PAA0

  ROM REVISION - 0001  WCS REVISION - 0001

  Test 1:  SEND DATA TEST, WITH OFFSET COMBINATIONS
  Test 2:  REQUEST DATA TEST, WITH OFFSET COMBINATIONS
  Test 3:  INVALIDATE TRANSLATION CACHE TEST
  Test 4:  SNDMDAT TEST, ENABLED/MAINTENANCE STATE
  Test 5:  SNDMDAT TEST, ENABLED STATE
  Test 6:  REQMDAT TEST, ENABLED/MAINT STATE
  Test 7:  REQMDAT TEST, ENABLED STATE
  Test 8:  SEND RESET TEST IN ENABLED STATE
  Test 9:  QUEUE CONTENTION TEST
  Test 10: BUFFER READ ACCESS TEST
  Test 11: BUFFER WRITE ACCESS TEST
  Test 12: WRITE TO GLOBAL BUFFER TEST
... End of run, 0 errors detected, pass count is 1,
  time is 15-JUL-1985 00:00:00.00
```

Example 3-8 Trace Printout For Functional Diagnostic EVGAB

3.4 REFERENCES

Table 3-3 Summary of the Functions of VAXcluster System Maintenance and Management Tools

Tool	Function
CI Exerciser	A Level 2R multipurpose exerciser that provides local CI interface functional testing as well as a means to determine the ability of VAXcluster nodes to reliably communicate using the CI bus.
VAXsim	A VAX system integrity monitor utility program that monitors and filters errors as they are logged by the VMS operating system. It provides the user with a warning mechanism that quickly identifies an option that is either failed or has degraded operationally. See Note 1.
SHOW Cluster	Allows the display of a large variety of utility information relevant to the configuration and operation of the VAXcluster of which the host system is a member. See Note 2.
SET HOST/HSC	Allows a terminal on a host VMS system to effectively become an HSC50 terminal. The user may then issue any standard HSC50 commands and look at or control the HSC50 just as if it were a terminal connected directly to one of the HSC50 terminal ports. See Note 3.

NOTES:

1. For more information, consult the *VAX System Integrity Monitor Manual.*
2. For more information, consult the *VAX/VMS Show Cluster Utility Manual.*
3. For more information, consult the *VAX/VMS DCL Dictionary* under SET HOST/HSC.

CHAPTER 4
CIBCA TROUBLESHOOTING

4.1 INTRODUCTION
This chapter provides information that will help in troubleshooting a CIBCA.

4.2 CIBCA MICROCODE REVISIONS
The loadable binary file CIBCA.BIN revision can be checked by using the VMS DUMP utility. A dump of the file is illustrated in Example 4-1.

After successful completion of CIBCA self-test, the microcode CIBCA.BIN is loaded into the control store. Control store location bb+108C contains the revision number of the functional microcode and location bb+1090 contains the revision number of the EEPROM code.

```
Dump of file DISK$BCA_LATEST:[BCA_LATEST.UCODE]CIBCA.BIN;2 on  5-DEC-1986 09:24:22.37
File ID (176,48192,0)   End of file block 65 / Allocated 66

Virtual block number 1 (00000001), 512 (0200) bytes                    <-- EEPROM VERSION NUMBER

00000000 00010000 1FFE0000 00015012 E01C3130 3030204E 49422E41 43424943 CIBCA.BIN 0001.^.P....^..... 000000
00000000 00000000 00000000 00000000 00C00000 00000000 00000000 00000000 ................................ 000020
00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 ................................ 000040
00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 ................................ 000060
00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 ................................ 000080
00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 ................................ 0000A0
00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 ................................ 0000C0
00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 ................................ 0000E0
00000000 00000000 00000000 00000001 00003000 00000011 58906A8F 31303030 0001.j.C.....0................ 000100
00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 ................................ 000120
                                                                       ^
                                                                       |
                                                                       FUNCTIONAL VERSION NUMBER
```

Example 4-1 CIBCA.BIN

4.3 CIBCA FRU LIST
See Table 1-1 for a list of the hardware FRUs used in the CIBCA.

4.4 CIBCA SELF-TEST TEST DESCRIPTION
The CIBCA self-test is made up of a collection of tests. These tests, along with a brief description of what each test checks, are listed in Table 4-1.

4.5 CIBCA SELF-TEST TEST/FAILURE CODE REGISTER (STFCR) bb+1FFC
Figure 4-1 illustrates the STFCR register. At the beginning of each test, register STFCR is loaded with the test number and 0 for a failure code. If test 1 is ready to run, the STFCR is loaded with 1000 (hex). During each key failure point of the test, the failure code is incremented in case of a failure. If a failure occurs, the STFCR is written to local store location 3FF (hex) or bb+1FFC (hex) so it can be accessed from the host bus.

Table 4-1 CIBCA Self-Test

Test Number	Test Name	Description
1	29116 Status Register Test	Checks the Z, N, O, and C bits.
2	ALU RAM Test	Checks the 32 RAM locations.
3	Index Register, Literal Register, Local Store, and MUX Test	Checks the different ways to WRITE and READ the control store.
4	Packet Buffer WRITE/READ Test	WRITEs all four packet buffers with a data pattern and then reads/checks the data.
5	Move Data Test	Performs a loopback of data from the BCAI DMA files through the link/packet buffer module and back into the BCAI DMA files.
6	(BIIC) STS Bit Set Test	Verifies that the BIIC self-test passed.
7	Data Mover WRITE Loopback to Local Store Test	Checks the SEL logic.
8	WRITE/READ Loopback to Local Store Test	Checks the IB, II, BCAI, BIIC, local store, and the 29116 data and address lines.
9	Toggle Register Test	Sets and clears the bits in the toggle register.
A	Parity Bits Test	Checks the different PE bits in the PMCSR register.
B	XBUS Register Test	WRITEs all the XBUS registers and then checks the data by way of the index register.
C	BI WRITE/READ Test	Performs a self-directed WRITE/READ to the BIIC GPR number 0 (defined in CIBCA as the PQBBR) to ensure that the CIBCA can access itself over the BI bus.

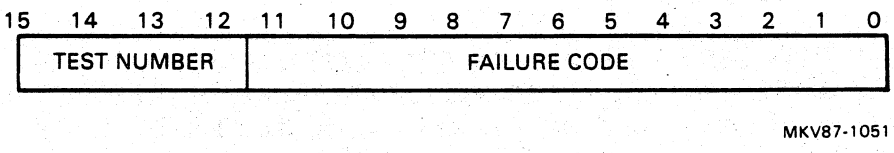


Figure 4-1 Self-Test Failure Code Register (STFCR)

4.6 CIBCA SELF-TEST FAILURE CODES AND FRUs

The following is a look-up chart for the STFCR register. If the CIBCA self-test fails, the contents of the STFCR register, bb+1FFC, in conjunction with this chart, can be used to identify the failing FRU.

STFCR Contents	Probable FRU Failure
100X	T1015 PCM
200X	T1015 PCM
3XXX	T1015 PCM
400X	T1015 PCM
5001	T1015/T1025 PCM/CCI
5002	T1015 PCM
5003	T1015 PCM
5004	T1015 PCM
5005	T1015 PCM
5006	T1015 PCM
5007	T1025 CCI
5008	T1025 CCI
5009	T1015 PCM
500A	T1025 CCI
500B	T1025 CCI
500C	T1025 CCI
500D	T1015 PCM
500E	T1015 PCM
500F	T1015 PCM
501X	T1015/T1025 PCM/CCI
5020	T1015 PCM
5021	T1015 PCM
5022	T1015 PCM
5023	T1015 PCM
5024	T1015/T1025 PCM/CCI
5025	T1025 CCI
5026	T1015 PCM
5027	T1015 PCM
5028	T1025 CCI
5029	T1025 CCI
502A	T1025 CCI
502B	T1015 PCM
502C	T1015 PCM
502B	T1015 PCM
502C	T1015 PCM
502D	T1015 PCM
502E	T1015 PCM
502F	T1015/T1025 PCM/CCI
503X	T1015/T1025 PCM/CCI
600X	T1015 PCM
700X	T1015 PCM
800X	T1015 PCM
90XX	T1015 PCM
A0XX	T1015 PCM
B0XX	T1015 PCM
C0XX	T1015 PCM

APPENDIX A
CIBCA REGISTER SUMMARY

A.1 INTRODUCTION

This section presents the interface conventions which allow programmer access to the CIBCA adapter functions and access to the software registers that are used to control and monitor the operation within the CIBCA adapter itself. Entry to these registers is accomplished through the VAXBI address space area.

A.2 VAXBI PHYSICAL ADDRESS SPACE

The physical address on the VAXBI is 30 bits long, thereby, providing a VAXBI physical address space of one gigabyte. A program will access this physical address space whenever it makes reference to a CIBCA adapter's hardware or software registers.

The VAXBI physical address space is divided into two parts; memory space and I/O space. Selection of memory space and I/O space is determined by address bit <29> of a READ or WRITE VAXBI bus transaction.

- Physical Memory Space Addresses

  The first 512 Mbytes (addresses 0000 0000 through 1FFF FFFF hexadecimal) are physical memory space addresses.

- I/O Space Addresses

  The last 512 Mbytes of the VAXBI physical address space (addresses 2000 0000 through 3FFF FFFF hexadecimal) are I/O space addresses.

Figure A-1 illustrates the physical partitioning of the VAXBI physical address space. As shown in Figure A-2, the 512 Mbyte VAXBI I/O address space is organized into several categories: map window, broadcast space, and node space. Only the VAXBI node space is used by the CIBCA adapter.

Node Space

The VAXBI node space (Figure A-3) is organized into sixteen 8-Kbyte address blocks. The CIBCA adapter hardware is assigned to one of these address blocks. This address block is referred to as the CIBCA adapter node. It is accessed whenever a CIBCA adapter hardware or software register is referenced.

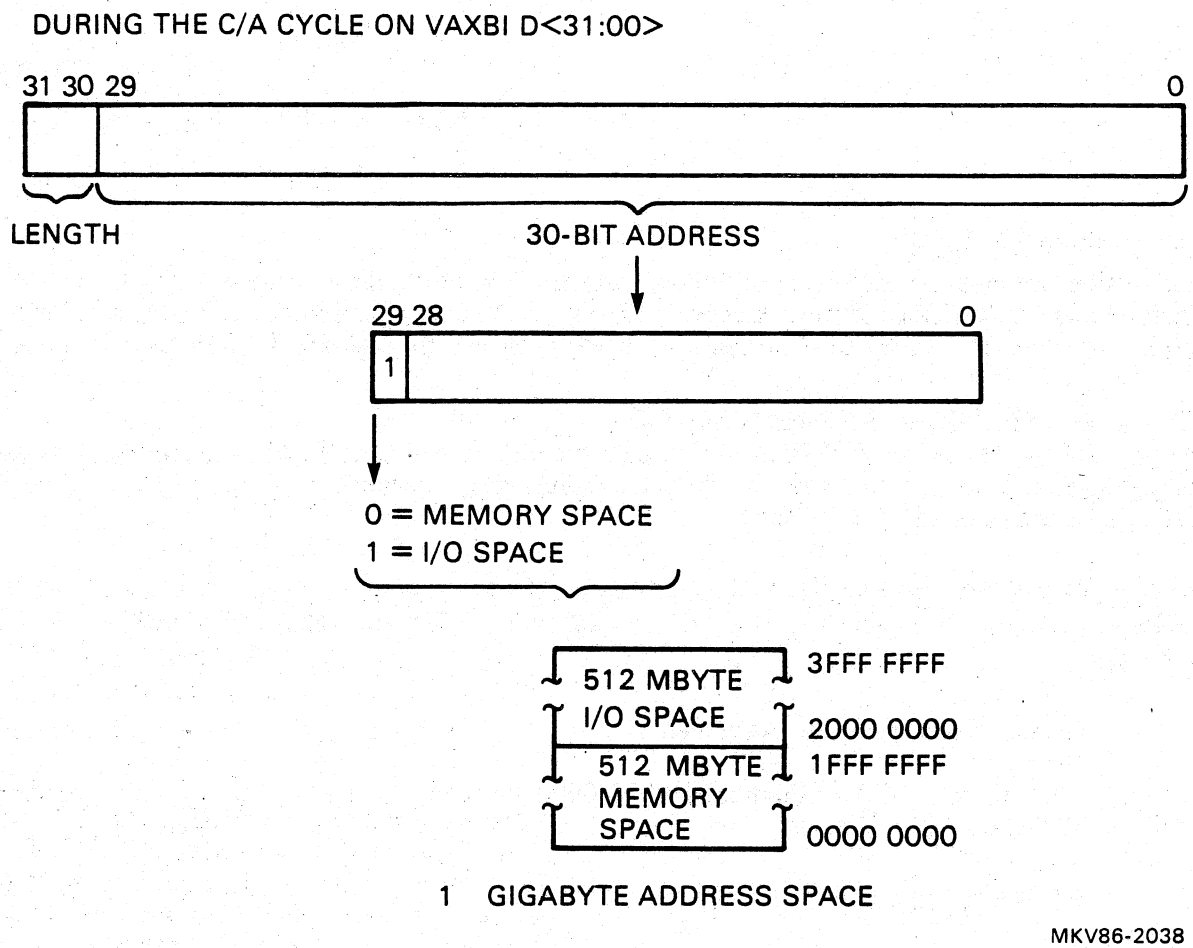


Figure A-1 VAXBI Physical Address Space

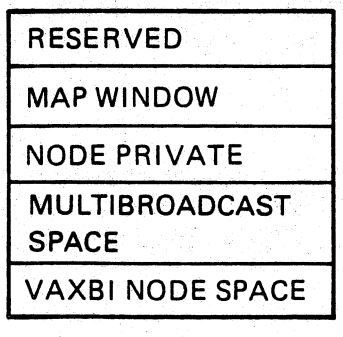


Figure A-2 VAXBI Physical I/O Address Space

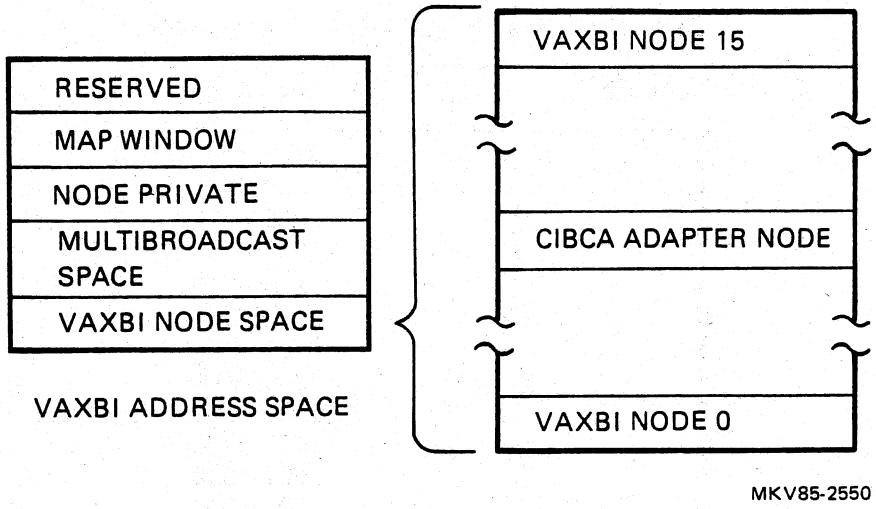


Figure A-3 VAXBI Node Space

A.3 CIBCA ADAPTER NODE

A.3.1 Addressing

The address area of the CIBCA adapter node is calculated by taking the base address representing the VAXBI I/O address space (2000 0000 hex) and adding 8K times the node ID, plus the offset address of the device register. For simplicity, this calculated address is represented by "bb+" whenever a reference is made to any of the following register bit maps. Figure A-4 illustrates the format structure of a 30-bit I/O address. Table A-1 lists the starting addresses of the 16 VAXBI node spaces.

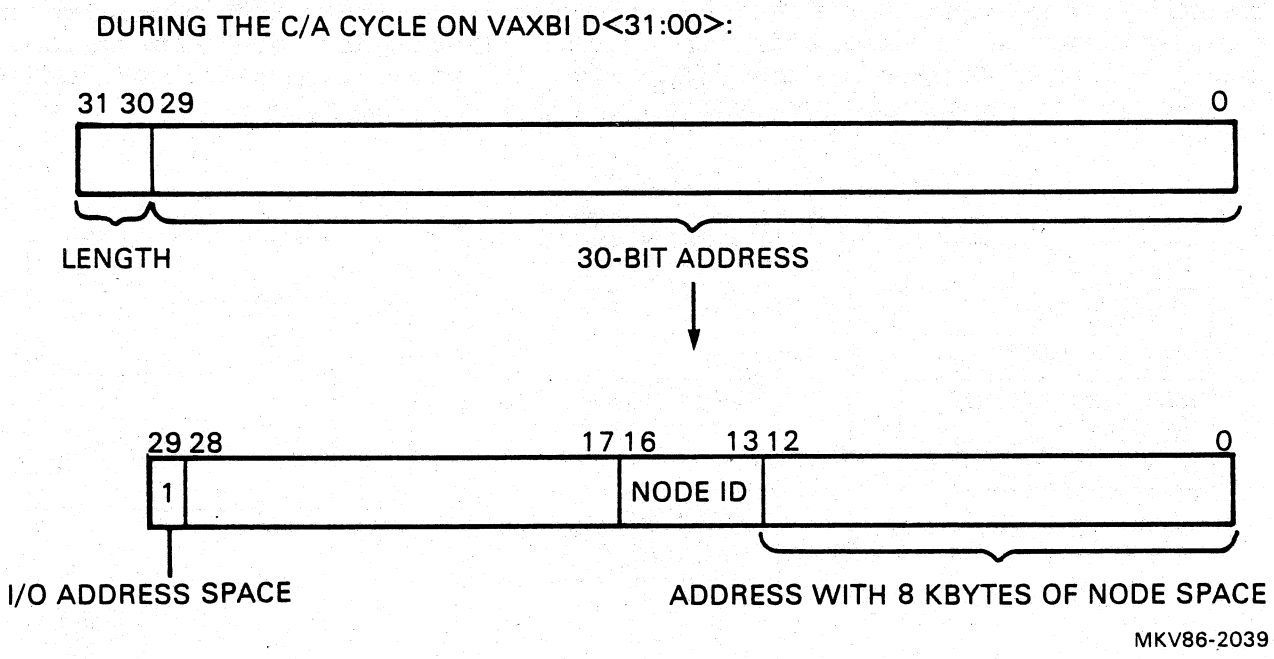


Figure A-4 30-Bit I/O Address Bit Map

Table A-1 Node Space Address Assignments

Node ID	Base Address
0	2000 0000
1	2000 2000
2	2000 4000
3	2000 6000
4	2000 8000
5	2000 A000
6	2000 C000
7	2000 E000
8	2001 0000
9	2001 2000
A	2001 4000
B	2001 6000
C	2001 8000
D	2001 A000
E	2001 C000
F	2001 E000

NOTE:
Offset address range 0000 to 1FFF hex.

A.3.2 Partitioning

As shown in Figure A-5, the CIBCA node space is divided into two segments: VAXBI CSR space and User CSR space. The VAXBI CSR space occupies the first 256 byte locations and is used by the VAXBI protocol and VAXBI control logic of the CIBCA adapter hardware. The User CSR space occupies the remaining locations of the address block. Only a portion of these addresses are used by the CIBCA adapter. Reading or writing to an unused register address will produce unpredictable results such as, unpredictable data, possible parity errors, and possible VAXBI NO ACK responses.

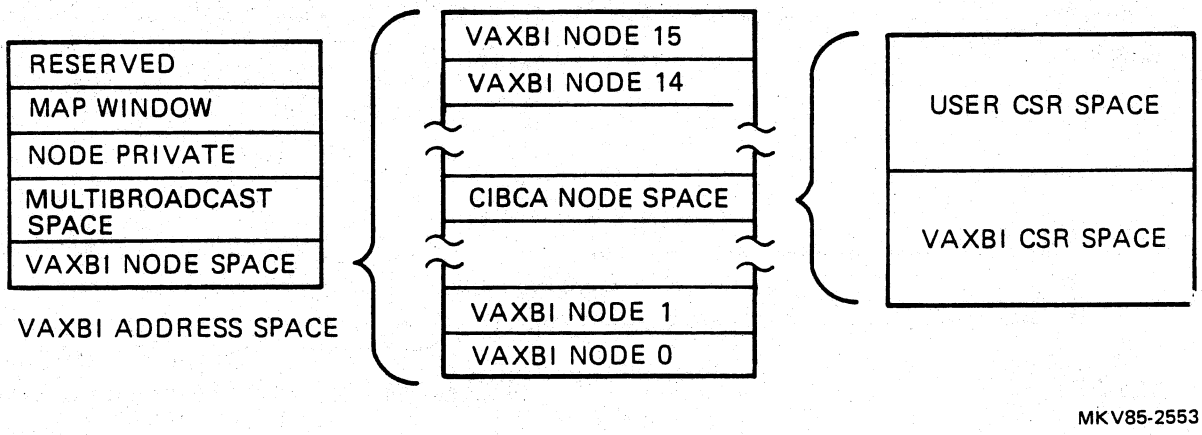


Figure A-5 CIBCA Address Node Space

A.3.3 Registers

Figure A-6 shows that the first 256 bytes of the CIBCA node space are reserved for the VAXBI CSR registers. VAXBI required registers and specific device registers fall into the category of VAXBI CSR registers. The VAXBI required registers are used by all VAXBI nodes including the CIBCA. The specific device registers are special-purpose VAXBI registers used to control the VAXBI device window area, and VAXBI data transfer control and interrupt control. The remaining addresses of the CIBCA node space are reserved for User CSR registers. The adapter registers fall into this category and are used for initializing and controlling the CIBCA adapter hardware. All these registers are accessed using longword addresses.

Figure A-7 illustrates the VAXBI interface registers and adapter registers.

NOTE

The CIBCA adapter hardware only issues longword or octaword VAXBI bus transactions.

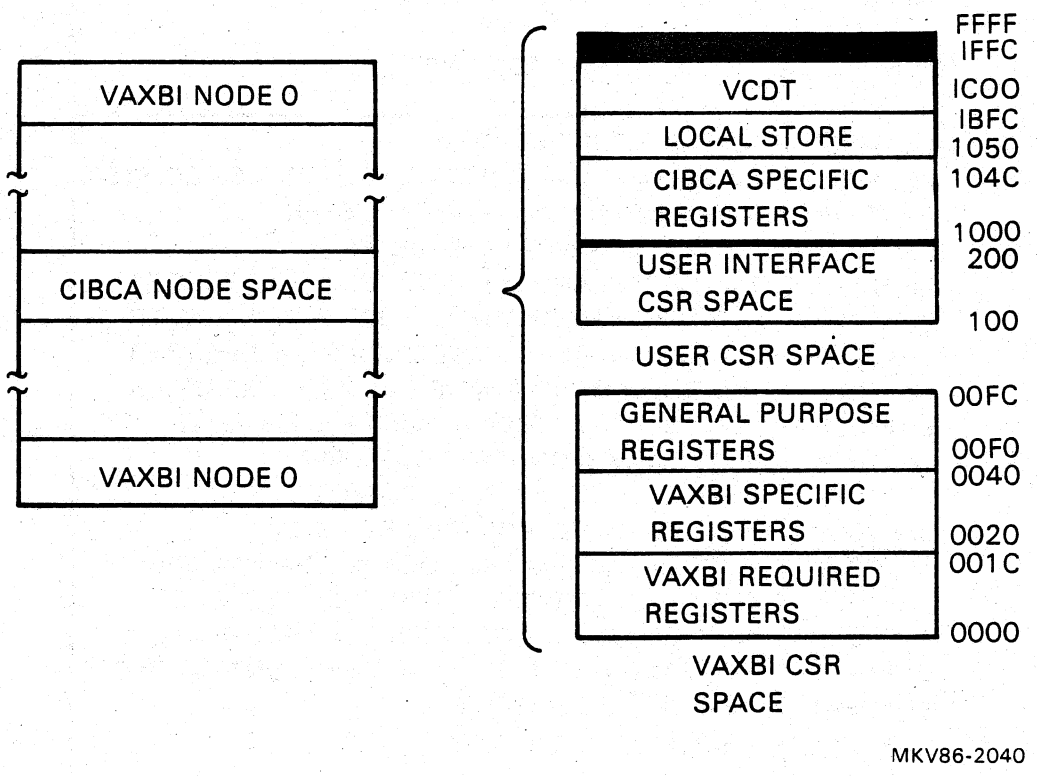


Figure A-6 CIBCA Adapter Register Address Space

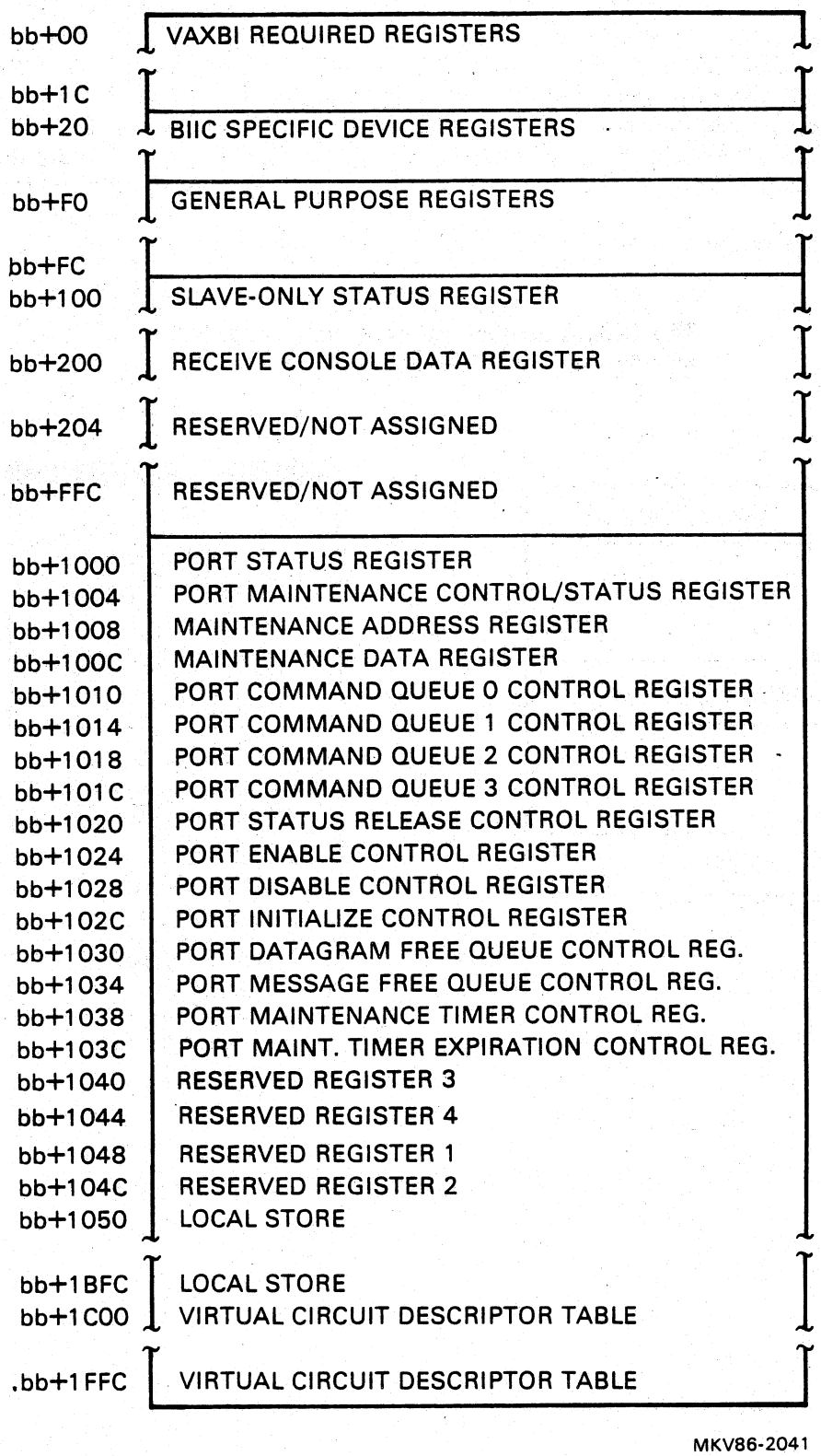


Figure A-7 VAXBI Interface Registers and Adapter Registers

A.4 VAXBI REQUIRED REGISTERS

Figure A-8 illustrates a more detailed diagram of the VAXBI required registers.

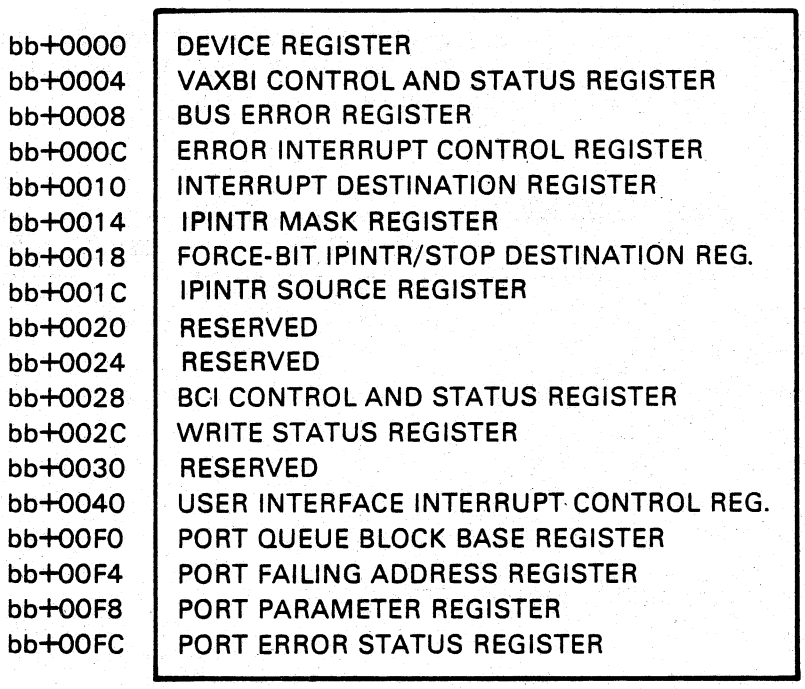


Figure A-8 VAXBI CSR Space

A.4.1 Device Register (DTYPE) (R/W,DMW,DCLOL) OFFSET = 0000

The Device Register (Figure A-9) address offset = 00 hex, field bits <15:00> identifies the type of node for use by the VMS operating system's device driver software. The device type assigned to the CIBCA adapter is 0108 (hex).

Field bits <23:16> identify the port revision level. Field bits <28:24> identify the link revision level. Bit <29> is reserved. Bit <30> indicates packet buffer size (0=1K, 1=4K). If bit <31> is a 0, it implies half-duplex operation. If bit <31> is a 1, it implies full-duplex operation.

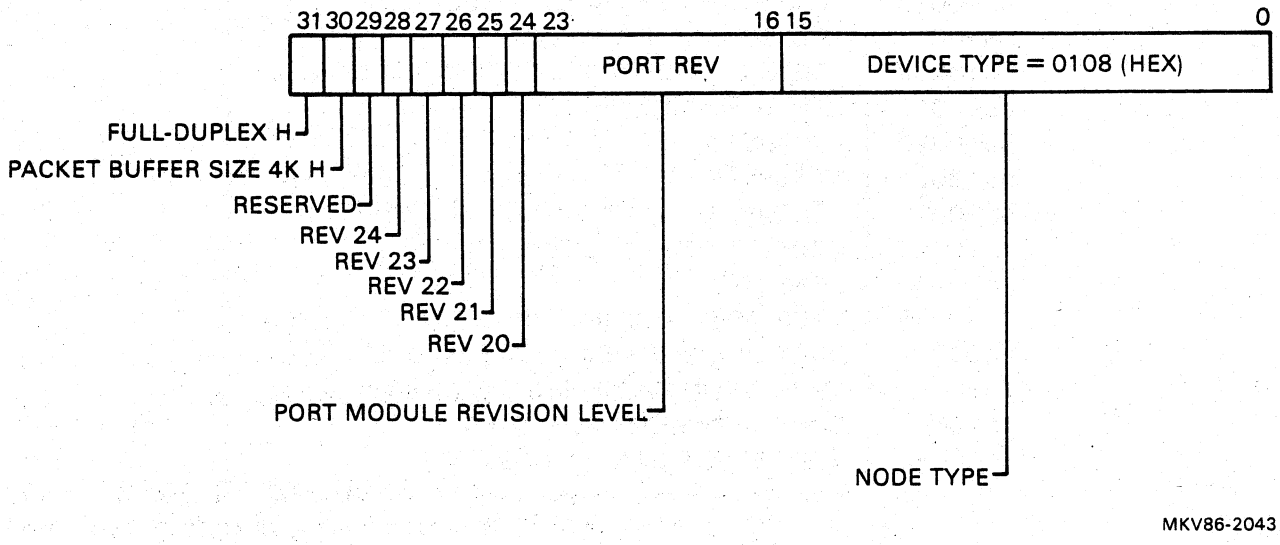


Figure A-9 Device Register Bit Map

A.4.2 VAXBI Control and Status Register (VAXBICSR) OFFSET = 0004

The VAXBI Control and Status Register, address offset 0004 (hex), contains control and status information bits. It also contains the BIIC type and the node ID, and specifies the mode of arbitration. Figure A-10 illustrates the register format. The bit assignments are described in Table A-2.

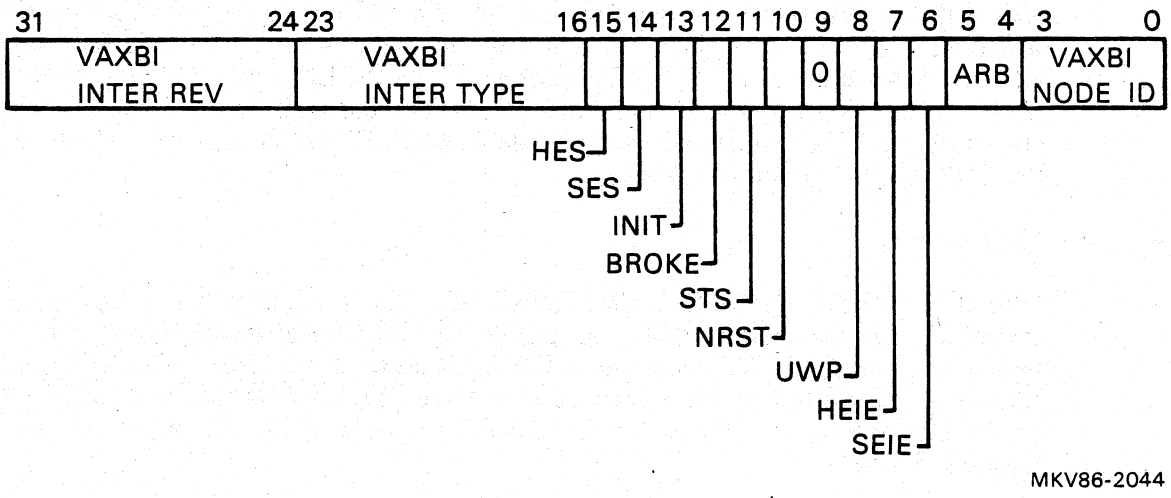


Figure A-10 VAXBI Control and Status Register

Table A-2 VAXBI Control and Status Register Bit Definitions

Bit	Description
<31:24>	VAXBI INTERFACE REVISION (RO) – Indicates revision level of the BIIC chip.
<23:16>	VAXBI INTERFACE TYPE (RO) – Indicates the type of device that provides the primary interface to the VAXBI (always 00000001).
<15>	HARD ERROR SUMMARY (RO) – Indicates that one or more of the hard error bits in the Bus Error Register are set.
<14>	SOFT ERROR SUMMARY (RO) – Indicates that one or more of the soft error bits in the Bus Error Register are set.
<13>	INIT – This bit is not used.
<12>	BROKE (W1C, DCLOS) – Self-test failure. Adapter will clear this bit when both the BIIC's internal self-test and the port's self-test passes. The port will do a self-test on power up only.
<11>	SELF TEST STATUS (R/W, DCLOC) – This bit will be a "1" if the BIIC's internal self-test passes. This bit enables the BIIC's BI drivers, and, therefore, a chip that fails self-test will be unable to drive the BI. If the node has a reset STS bit, then a WRITE that sets this bit will receive a NOACK response. Because the node's VAXBI driver is disabled, the WRITE must be either a loopback or a VAXBI internode transaction.

Table A-2 VAXBI Control and Status Register Bit Definitions (Cont)

Bit	Description
<10>	NODE RESET (SC) – Writing a "1" to this bit location forces the initiation of a complete node self-test. When this bit is written as a "1", the self-test status (STS) bit must also be written as a "1" to ensure proper operation of the WRITE-type transaction. READs to this bit will always return a "0". The BIIC asserts the BCI DC LO L line following the setting of the NRST bit. When BCI DC LO L line is deasserted, the BIIC begins its self-test. This will also cause the CIBCA to reload self-test code from the EEPROM into the control store and the CIBCA self-test will commence.
<09>	Must be zero.
<08>	UNLOCK WRITE PENDING (W1C, DCLOC, SC) – Indicates that a successful IRCI transaction has been completed by the master port interface at this node and there has not been a subsequent UMWCI command. This bit is cleared by a UMWCI transaction that is completed successfully by the master port interface. If a UWMCI transaction is attempted by the master port interface when the UWP bit is not set, the ISE bit in the Bus Error Register will be set.
<07>	HARD ERROR INTERRUPT ENABLE (R/W, DCLOC, STOPC) – Enables an error interrupt to be generated by the VAXBI node when HES is asserted. This bit should be "0" for the CIBCA.
<06>	SOFT ERROR INTERRUPT ENABLE (R/W, DCLOC, STOPC, [VMSL]) – This bit determines whether an error interrupt will be generated by this node when SES is asserted. VMS can set this bit to allow an interrupt to be generated by this node when a soft error is detected. VMS must load the vector in the Error Interrupt Control Register if it sets this bit.
<05:04>	ARBITRATION (R/W, DCLOC) – Two arbitration control bits determine the mode of arbitration to be used by the interface.

ARB 1 0	Arbitration Mode
0 0	Dual Round Robin
0 1	Fixed High Priority (Reserved)
1 0	Fixed Low Priority (Reserved)
1 1	Disable Arbitration (Reserved)

<03:00>	VAXBI NODE ID (RO, DMW, DCLOL) – Indicates this node ID which is formed by backplane jumpers on pins B9, B10, B11, and B12 on the BI backplane. This information is loaded from BCI I <3:0> H lines during the last cycle in which BCI DC LO is asserted.

A.4.3 Bus Error Register (BER) (W1C,DCLOC) OFFSET = 0008

The Bus Error Register provides bus error status information resulting from VAXBI bus or internal loopback transactions. Figure A-11 illustrates the register format. The bit assignments are described in Table A-3.

NOTE

Unless otherwise noted, all BER bits can be set during VAXBI and loopback transactions. Bits <30:16> are hard error bits, and bits <2:0> are soft error bits. Bit <3>, user parity enable (UPEN), is not an error bit. It indicates the BIIC parity mode.

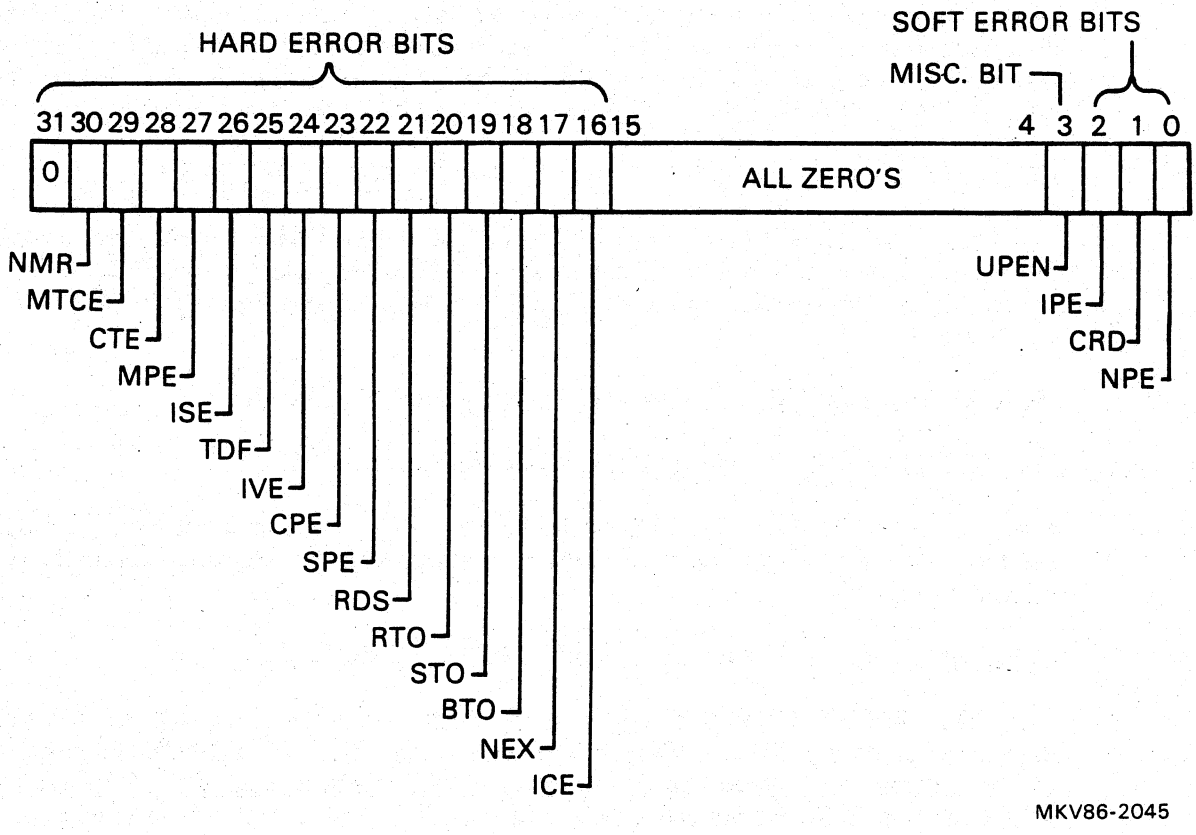


Figure A-11 Bus Error Register

Table A-3 Bus Error Register Bit Definitions

Bit	Description
<31>	Will be zero.
<30>	NO ACK to MULTI-RESPONDER COMMAND RECEIVED (W1C, DCLOC) – This bit will be set if the master receives a NO ACK command response for an INVAL, STOP, INTR, IPINTR, BDCST, or RESERVED COMMAND.
<29>	MASTER TRANSMIT CHECK ERROR – During cycles of a transaction in which the master is the only source of data on the BI D, I, and P lines, the BIIC verifies that the master's transmitted data matches the received data from the BI. If the transmitted data does not match the received data, this bit is set. This check is not performed for the master's assertion of its encoded ID on the I lines during the embedded ARB cycle. When this bit sets, the MSE in the Port Status Register also sets, causing the port to initiate an interrupt.
<28>	CONTROL TRANSMIT ERROR – This bit is not used by the CIBCA.
<27>	MASTER PARITY ERROR – This bit is set if the master detects a parity error on the bus during a data cycle of a transaction that has an ACK confirmation on the CNF <2:0> lines. When this bit sets, the MSE in the Port Status Register also sets, causing the port to initiate an interrupt.
<26>	INTERLOCK SEQUENCE ERROR – Not used by the CIBCA.
<25>	TRANSMITTER DURING FAULT – Not used by the CIBCA.
<24>	IDENT VECTOR ERROR – This bit is set if an ACK response is not received from the master. When this bit sets, the MSE in the Port Status Register also sets, causing the port to initiate an interrupt.
<23>	COMMAND PARITY ERROR – Not used by the CIBCA.
<22>	SLAVE PARITY ERROR – This bit is set by the selected slave if the BIIC detects a parity error during a data cycle of a WRITE-type transaction. The BIIC suppresses all parity error checking during data cycles that do not have an ACK confirmation on the CNF lines. This assures that the BIIC will not check parity during data cycles that have undefined data, such as STALLed data cycles. When this bit sets, the MSE in the Port Status Register also sets, causing the port to initiate an interrupt.
<21>	READ DATA SUBSTITUTE – This bit is set if a read data substitute (RDS) or reserved status code is received during a READ-type or IDENT (for vector status) transaction. In order for this bit to be set the BIIC logic also requires a successful parity check for the data cycle that contains the RDS code. This bit will be set even if the transaction is aborted some time after the receipt of the RDS or reserved status code. When this bit sets, the MSE in the Port Status Register also sets, causing the port to initiate an interrupt.
<20>	RETRY TIMEOUT – This bit is set if the master receives 4096 consecutive RETRY responses from the selected slave for the same master port transaction. When this bit sets, the MSE in the Port Status Register also sets, causing the port to initiate an interrupt.

Table A-3 Bus Error Register Bit Definitions (Cont)

Bit	Description
<19>	STALL TIMEOUT – This bit is set if the slave port asserts the STALL code on the RS<1:0> lines for 128 consecutive cycles. When this bit is set, the MSE in the Port Status Register also sets, causing the port to initiate an interrupt.
<18>	BUS TIMEOUT – This bit is set if the BIIC is unable to start at least one pending transaction before 4096 cycles have elapsed. When this bit is set, the MSE in the Port Status Register also sets, causing the port to initiate an interrupt.
<17>	NON-EXISTENT ADDRESS – This bit is set when a NO ACK response is received for a READ-type or WRITE-type command sent by the BIIC. This bit is set only if the master loopback and master parity check of the command/address cycle was successful. This bit is not set for NO ACK responses to other commands. When this bit is set, the MSE in the Port Status Register also sets, causing the port to initiate an interrupt.
<16>	ILLEGAL CONFIRMATION ERROR – A reserved or illegal CNF code has been received during a transaction in which the BIIC is involved. This bit can be set by either the master or slave node. NO ACK is not considered an illegal response for command confirmation. When this bit is set, the MSE in the Port Status Register also sets, causing the port to initiate an interrupt.
<15:04>	All zeros.
<03>	USER PARITY ENABLE (RO, DCLOC) – This bit indicates the BIIC parity mode. A "1" indicates the BIIC is configured for user-generated parity, while a "0" indicates the BIIC will provide the parity generations. This is opposite to the polarity provided on the BCI P0 line during BI DC L, which indicates which device generates parity. On power-up, an "H" (default) configures the BIIC for BIIC-generated parity, whereas, an "L" configures the chip for user-generated parity. While UPEN is set, the user interface is required to provide parity on the BCI P0 L line whenever BCI SDE L or BCI MDE L is asserted. The CIBCA sets this bit at the end of a successful self-test but before it clears the BROKE bit in the VAXBI Control and Status Register. The port controller module provides parity for the BIIC.
<02>	ID PARITY ERROR (W1C, DCLOC) – This bit is set if a parity error is detected on the BI I lines when the master's encoded ID is asserted during embedded ARB cycles. All nodes perform this parity check. This bit is not set during loopback request transactions. When this bit is set, it causes the BIIC to generate an interrupt if the SEIE bit in the VAXBI Control and Status Register is set.
<01>	CORRECTED READ DATA (W1C, DCLOC) – This bit is set if the master receives a corrected read data status code. For this bit to be set, the BIIC logic requires the receipt of good parity for the data cycle that contains the CRD code. This bit is set even if the transaction aborts after the CRD status code has been received. When this bit is set, it causes the BIIC to generate an interrupt if the SEIE bit in the VAXBI Control and Status Register is set.
<00>	NULL BUS PARITY ERROR (W1C, DCLOC) – Odd parity is detected on the bus during the second cycle of a two-cycle sequence during which BI NO ARB L and BI BSY L were unasserted. When this bit is set, it causes the BIIC to generate an interrupt if the SEIE bit in the BI Control and Status Register is set.

A.4.4 Error Interrupt Control Register (EINTRCSR) OFFSET = 000C

The Error Interrupt Control Register controls the operation of the interrupts initiated by a BIIC detected bus error (which sets a bit in the Bus Error Register) or by setting the FORCE bit in this register. An error interrupt request is the logical OR of all the BER register bits with the FORCE bit and error interrupt enable bits set in the VAXBI Control and Status Register. This register is set up by the software if the SEIE bit in the VAXBI Control and Status Register is set.

Figure A-12 illustrates the register format. The bit assignments are described is Table A-4.

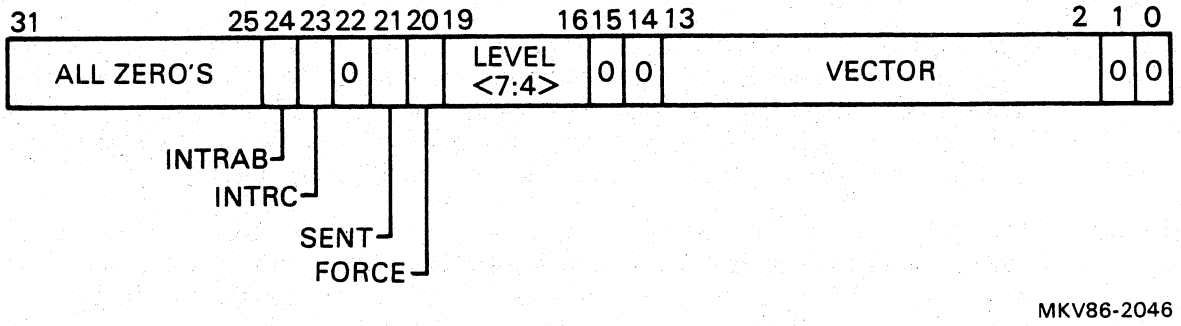


Figure A-12 Error Interrupt Control Register

Table A-4 Error Interrupt Control Register Bit Definitions

Bit	Description
<31:25>	All zeros.
<24>	INTERRUPT ABORT BIT - Not used by the CIBCA.
<23>	INTERRUPT COMPLETE BIT - Not used by the CIBCA.
<22>	Zero.
<21>	INTERRUPT SENT - Not used by the CIBCA.
<20>	INTERRUPT FORCE - Not used by the CIBCA.
<19:16>	LEVEL <7:4> (R/W, DCLOC) - Not used by the CIBCA.
<15:14>	Zero.
<13:02>	VECTOR (DCLOC, R/W, [VMSL]) - This field contains the vector used during error interrupt sequences. It is transmitted when this node wins an IDENT ARB cycle on an IDENT transaction that matches the conditions in the Error Interrupt Control Register.
<01:00>	Zero.

A.4.5 Interrupt Destination Register (INTRDES) (R/W,DCLOC,[VMSL]) OFFSET = 0010

The Interrupt Destination Register indicates which nodes of the VAXBI are to be targeted by interrupt commands. The destination is sent out during the INTR command and is monitored by all nodes to determine whether to respond. Figure A-13 illustrates the register format. The bit assignments are described in Table A-5.

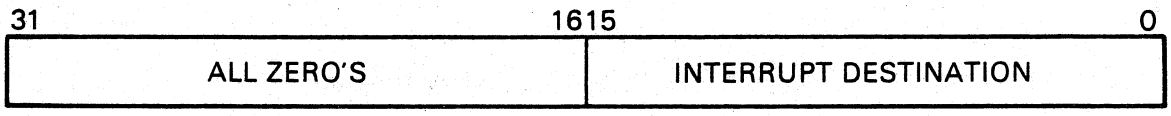


Figure A-13 Interrupt Destination Register

Table A-5 Interrupt Destination Register Bit Definitions

Bit	Description
<31:16>	All zeros.
<15:00>	INTERRUPT DESTINATION – This field determines which node(s) on the VAXBI are to be targeted by INTR commands sent by this node. This field is sent out during the INTR command and is used by the destination to determine whether to respond. During an IDENT command, the decoded master's ID is compared to the destination field. If there is no match, this node will not respond to the IDENT. If there is a match, the master's decoded ID is set in the Interrupt Destination Register. The BIIC will then respond to the IDENT, provided that there is an unserviced interrupt request at that node that matches the level transmitted in the IDENT command.

A.4.6 IP Interrupt Mask Register (IPINTRMSK) (R/W,DCLOC) OFFSET = 0014

The CIBCA does not use this register.

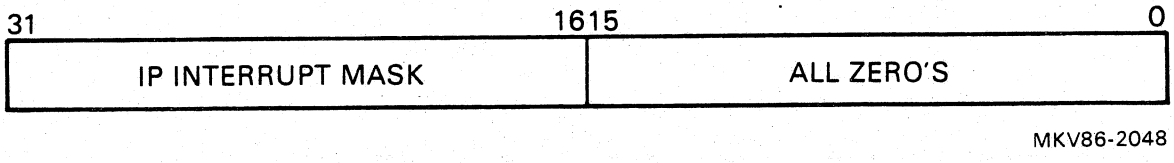


Figure A-14 IP Interrupt Mask Register

A.4.7 Force Bit IPINTR/STOP Destination Register (IPIDR) (R/W,DCLOC) OFFSET = 0018

The CIBCA does not use this register.

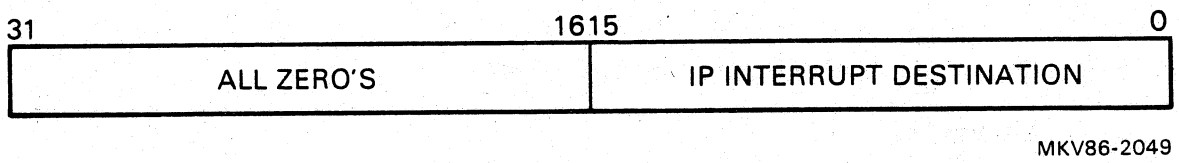


Figure A-15 Force Bit IPINTR/STOP Destination Register

A.4.8 IP Interrupt Source Register (IPNTRSRC) (W1C,DCLOC) OFFSET = 001C

The CIBCA does not use this register.

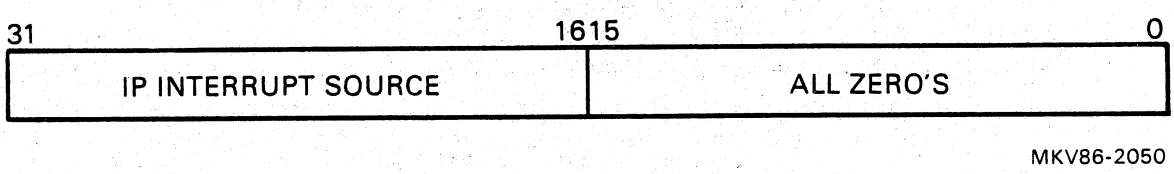


Figure A-16 IP Interrupt Source Register

A.4.9 Starting Address Register

The Starting Address Register is not used by the CIBCA.

A.4.10 Ending Address Register

The Ending Address Register is not used by the CIBCA.

A.4.11 BCI Control and Status Register (BCICSR) OFFSET = 0028

The BCI Control and Status Register enables various functions between the BIIC chip and the user's port to occur. Figure A-17 illustrates the register format. The bit assignments are described in Table A-6.

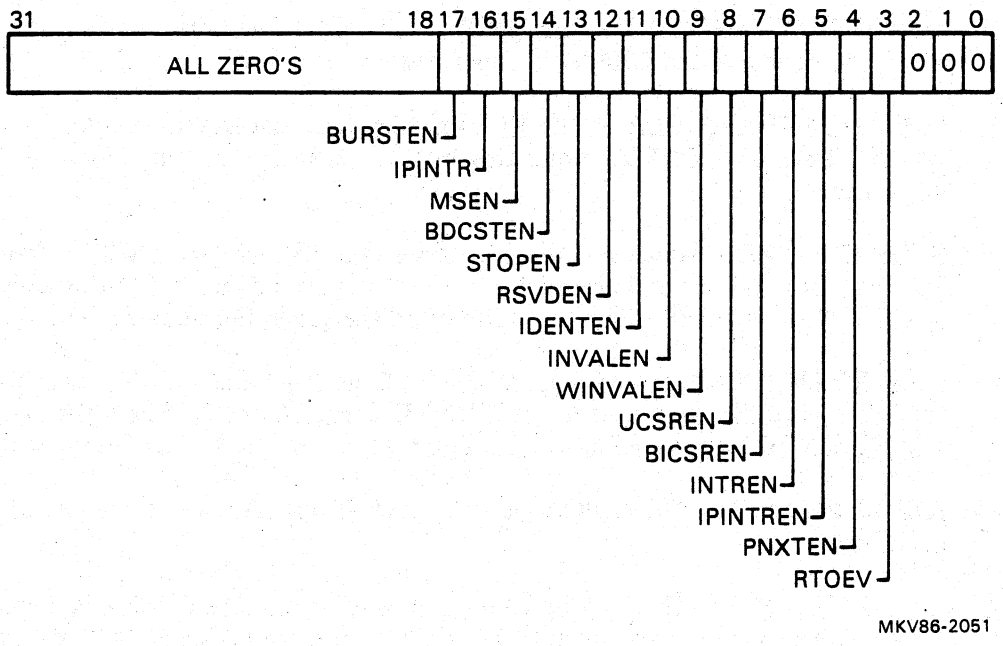


Figure A-17 BCI Control and Status Register

Table A-6 BCI Control and Status Register Bit Definitions

Bit	Description
<31:18>	All zeros.
<17>	BURST ENABLE – When set, this bit causes BI NO ARB L to be asserted continuously after the next successful ARB by this node, until the BURSTEN bit is reset or BCI MAB L is asserted. The assertion of BI MAB L does not reset the BURSTEN bit. It merely clears the burst mode state in the BIIC, which is holding BI NO ARB L. Unless a subsequent transaction clears this bit, the next successful ARB by this node will cause the BIIC to once again hold BI NO ARB L continuously. Only BI requests may be used in burst mode. Loopback requests must not be used. The CIBCA clears this bit. If this bit is set, the action of the CIBCA is undefined.
<16>	IP INTERRUPT/STOP FORCE (R/W, DCLOC) – When set, this bit causes the BIIC to arbitrate for the bus and transmit an IPINTR or STOP command. The command transmitted depends on the command stored in the Force Bit IPINTR/STOP Command Register, using the Force Bit IPINTR/STOP Destination Register for the destination field. The IPINTR/STOP Force Bit is reset by the BIIC following the transmission of an IPINTR transaction. If the transmission fails, the NICIPS (NO ACK or illegal CNF received for Force Bit INTR/STOP command) EV code is output and the NMR (NO ACK to Multiresponder Command Received) bit is set. The CIBCA clears this bit. If this bit is set, the action of the CIBCA is undefined.

Table A-6 BCI Control and Status Register Bit Definitions (Cont)

Bit	Description
<15>	MULTICAST SPACE ENABLE (R/W, DCLOC) – The CIBCA clears this bit. If this bit is set, the action of the CIBCA is undefined.
<14>	BROADCAST ENABLE – The BI BROADCAST command directed at the CIBCA is NO ACKED. The CIBCA clears this bit. If this bit is set, the action of the CIBCA is undefined.
<13>	STOP ENABLE – When set, this bit causes the BIIC to assert SEL and the appropriate SC<2:0> code following the receipt of a STOP command directed at this node. The CIBCA sets this bit at the end of self-test and before the green light comes ON.
<12>	RESERVED ENABLE – When set, the BIIC asserts SEL and the appropriate SC<2:0> code following the receipt of a RESERVED command code. The CIBCA clears this bit. The CIBCA NO ACKs RESERVED commands that are received even if the bit is set.
<11>	IDENT ENABLE – The CIBCA clears this bit. If this bit is set, the action of the CIBCA is undefined.
<10>	INVALIDATE ENABLE – The CIBCA clears this bit. The CIBCA NO ACKs INVALIDATE commands that are received. If this bit is set, the action of the CIBCA is undefined.
<09>	WRITE INVALIDATE ENABLE – The CIBCA clears this bit. If this bit is set, the action of the CIBCA is undefined.
<08>	USER INTERFACE CSR SPACE ENABLE – When set, this bit causes the BIIC to assert SEL and the appropriate SC<2:0> code following the receipt of a READ or WRITE type command directed at this node's User CSR space. The CIBCA sets this bit at the end of a successful self-test just after the green light comes ON to allow access to the port's User CSR space.
<07>	BIIC CSR SPACE ENABLE – The CIBCA clears this bit.
<06>	INTERRUPT ENABLE – The CIBCA clears this bit. If this bit is set, the action of the CIBCA is undefined. The CIBCA returns NO ACKs to BI INTERRUPT commands.
<05>	IP INTERRUPT ENABLE – The CIBCA does not respond to IP INTR commands. If this bit is set, the action of the CIBCA is undefined.
<04>	PIPELINE NEXT ENABLE – The CIBCA clears this bit. If this bit is set, the action of the CIBCA is undefined.
<03>	RTO EV ENABLE – The CIBCA sets this bit at the end of a successful self-test, but before it clears the BROKE bit in the VAXBI Control and Status Register. If this bit is cleared, the action of the CIBCA is undefined.
<02:00>	Zero.

A.4.12 Write Status Register (WSTAT) OFFSET = 002C

The CIBCA does not use this register.

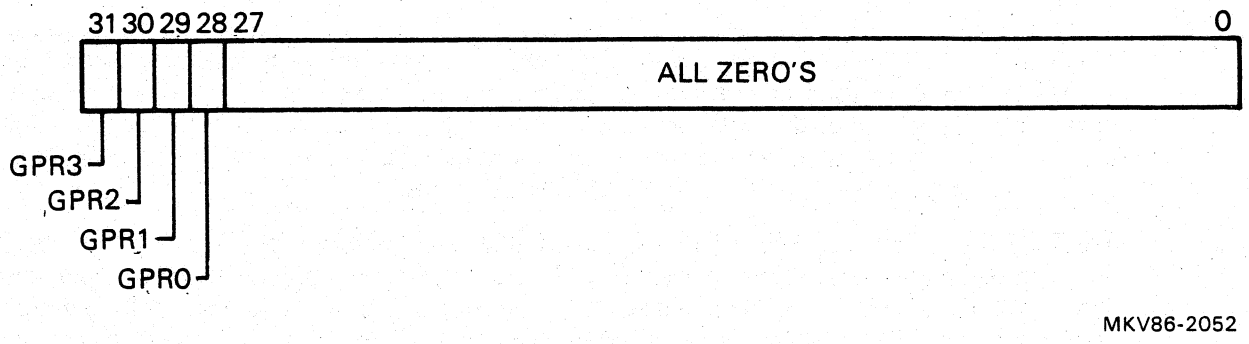


Figure A-18 Write Status Register

A.4.13 User Interface Interrupt Control Register (UINTRCSR) OFFSET = 0040

The User Interface Interrupt Control Register controls the operation of interrupts initiated by the user interface. Interrupts may be initiated by either the assertion of any of the BCI INT <7:4> L lines or by setting any of the force bits in this register. Figure A-19 illustrates the register format. The bit assignments are described in Table A-7.

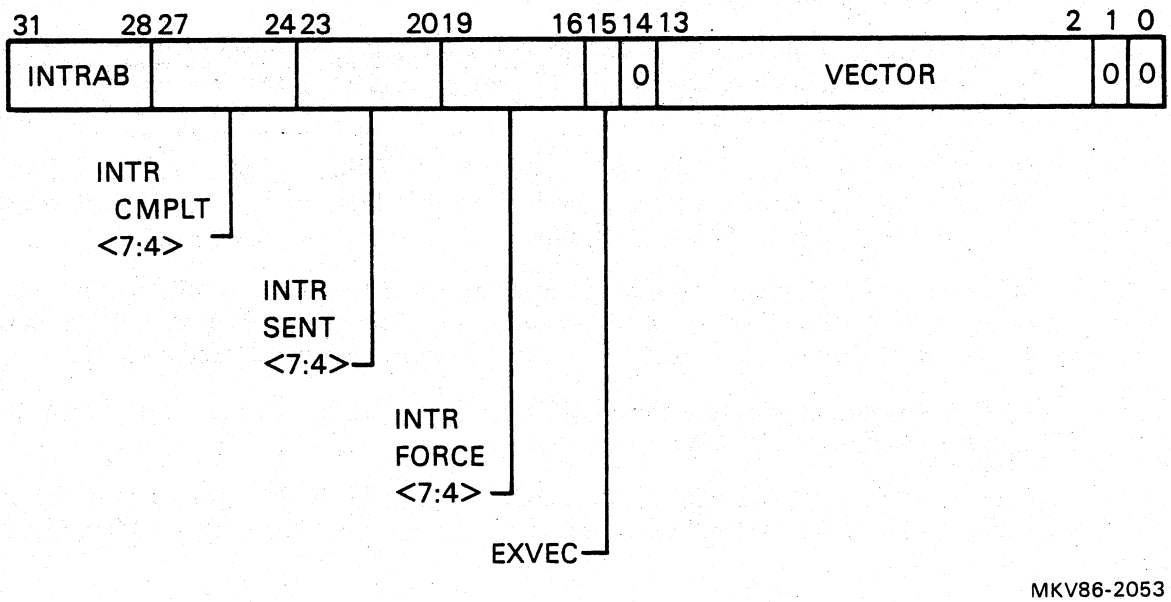


Figure A-19 User Interface Interrupt Control Register

Table A-7 User Interface Interrupt Control Register Bit Definitions

Bit	Description
<31:28>	INTERRUPT ABORT <7:4> (W1C, DCLOC) - There are four interrupt abort bits corresponding to the four interrupt levels. An interrupt abort bit is set if an INTR command sent under the control of this register is aborted. INTRAB is a status bit set by the BIIC and can be reset only by the user interface. The bit has no effect on the ability of the BIIC to send or respond to further INTR or IDENT transactions.
<27:24>	INTERRUPT COMPLETE <7:4> (W1C, DCLOC) - There are four interrupt complete bits corresponding to the four interrupt levels. An interrupt complete bit is set when the vector for an interrupt has been successfully transmitted, or if an INTR command sent under the control of this register is aborted. Removal of the interrupt request clears the corresponding INTRC bit. While an INTR bit is set, no further interrupts at that level are generated by this register. Further, no IDENTS will be responded to by this register when the INTRC bit is set at the IDENT level.
<23:20>	INTERRUPT SENT <7:4> (W1C, DCLOC, STOPC) - There are four interrupt sent bits corresponding to the four interrupt levels. When asserted, an INTR SENT bit indicates that an INTR command for the corresponding level has been successfully transmitted. This bit is cleared during an IDENT command following the detection of a level and master ID match. Clearing the bit allows the interrupt to be resent if this node loses the IDENT arbitration, or if the node wins but the vector transmission fails. Deassertion of an interrupt request causes the appropriate INTR SENT bit to be cleared. It is not necessary for the INTR SENT bit at a given level to be set in order for the BIIC to respond to an IDENT at that level. All that is required is that the interrupt request be posted at the IDENT level.
<19:16>	INTERRUPT FORCE <7:4> (R/W, DCLOC, STOPC) - There are four FORCE bits corresponding to the four interrupt levels. Setting a FORCE bit is equivalent to asserting the corresponding BCI INT<7:4> L input. When multiple interrupt requests are asserted simultaneously, the BIIC transmits INTR commands for the highest priority requests first. Similarly, when an IDENT command solicits more than one level, the BIIC responds with the highest pending level. CIBCA diagnostics may use the FORCE bits to cause the CIBCA to generate interrupt commands onto the VAXBI.
<15>	EXTERNAL VECTOR (R/W, DCLOC) - The CIBCA does not support the external vector mode. If the external vector bit is set, the action of the CIBCA is undefined.
<14>	Zero.
<13:02>	VECTOR (R/W, DCLOC, [VMSL]) - This field contains the vector used during user interface interrupt sequences (unless the external vector bit is set). The vector is transmitted when this node wins an IDENT ARB that matches the conditions in the User Interface Interrupt Control Register. The vector must be loaded by software prior to enabling interrupts. The interrupt level utilized by the CIBCA is Level 4.
<01:00>	Zero.

A.4.14 Port Queue Block Base Register (PQBBR) (R/W,DCLOC,[SC]) OFFSET = 00F0

The Port Queue Block Base Register contains the physical address of the base for the port queue block in bits <28:09>. All other bits must be zero.

The PQBBR is READ/WRITE by the port driver and can be written only when the port is in the disabled or disabled/maintenance state. Its value before being written is unpredictable. Figure A-20 illustrates the register format.

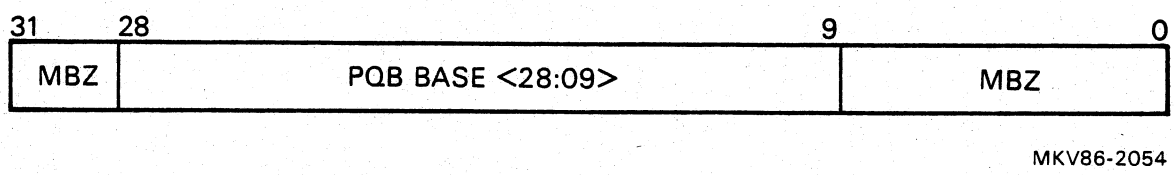


Figure A-20 Port Queue Block Base Register

A.4.15 Port Failing Address Register (PFAR) (R/W,DCLOC,[SC]) OFFSET = 00F4

For DSE interrupts the PFAR contains the virtual address or structure. For MSE interrupts and buffer memory system errors, the PFAR contains a physical address.

The PFAR is READ ONLY by the port driver and valid after a DSE or MSE interrupt, or after a response with buffer memory system error status. Figure A-21 illustrates the register format.

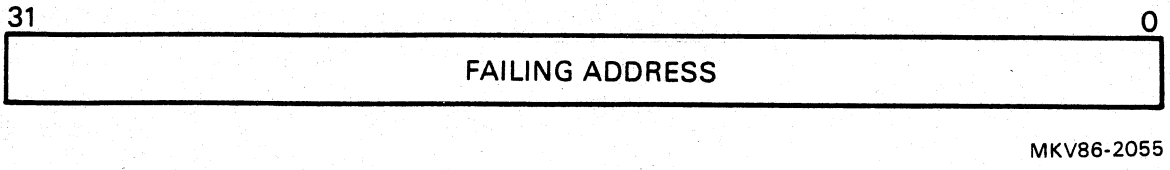


Figure A-21 Port Failing Address Register

A.4.16 Port Parameter Register (PPR) (R/W,DCLOC,[SC]) OFFSET = 00F8

The Port Parameter Register contains port implementation parameters and the port number. The PPR is set up by microcode during the port initialization process. It is valid in any state except the uninitialized state. The PPR is READ ONLY. Writing to this register destroys the port state with unpredictable results. Figure A-22 illustrates the register format. The bit assignments are described in Table A-8.

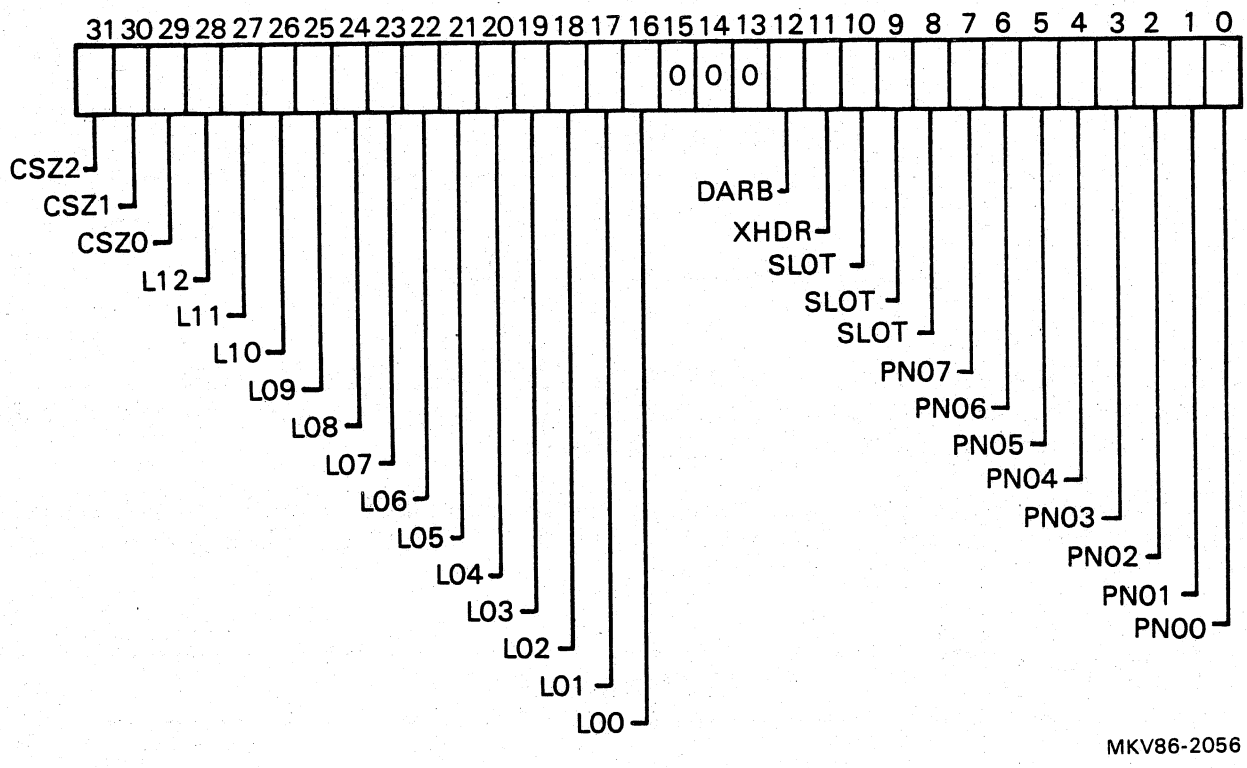


Figure A-22 Port Parameter Register

Table A-8 Port Parameter Register Bit Definitions

Bit	Description
<31:29>	CLUSTER SIZE <02:00> – This field indicates the maximum number of nodes allowed on the CI as follows:

CSZ 02	01	00	Cluster Size (decimal)	Range (decimal)
0	0	0	16 (Max)	0–15
0	0	1	32 (Max)	0–31
0	1	0	64 (Max)	0–63
0	1	1	128 (Max)	0–127
1	X	X	Reserved	

<28:16> LENGTH <12:00> – This field indicates the size of the internal buffers available for message and data transfers. 4 Kbyte buffers are utilized, therefore, this field is preset to FF8 (hex) or [4088 (dec)].

<15:13> Reserved and read as zero.

<12> DISABLE ARBITRATION – When this bit is set, defeats the normal arbitration sequence and allows the LINK to transmit after waiting only one basic quiet slot (Delta time).

<11> EXTENDED HEADER – When this bit is set, allows the LINK to extend the number of bit synchronous characters in the header.

<10:08> ALTER DELTA TIME <01:00> – These three bits force the LINK to a specific quiet slot Delta time as follows:

ADT<2>	ADT<1>	ADT<0>	QUIET SLOT COUNT (in decimal)
0	0	0	7
0	0	1	10
0	1	0	14
0	1	1	16
1	0	0	21
1	0	1	25
1	1	0	32
1	1	1	Illegal

<07:00> PORT NUMBER <07:00> – This field indicates the CI node number of this port.

A.4.17 Port Error Status Register (PESR) (R/W,DCLOC,[SC]) OFFSET = 00FC

The PESR indicates the type of error which resulted in a data structure error (PSR__DSE) interrupt. PESR is READ ONLY by the port driver and valid after a Port Status Register DSE interrupt. Figure A-23 illustrates the register format.

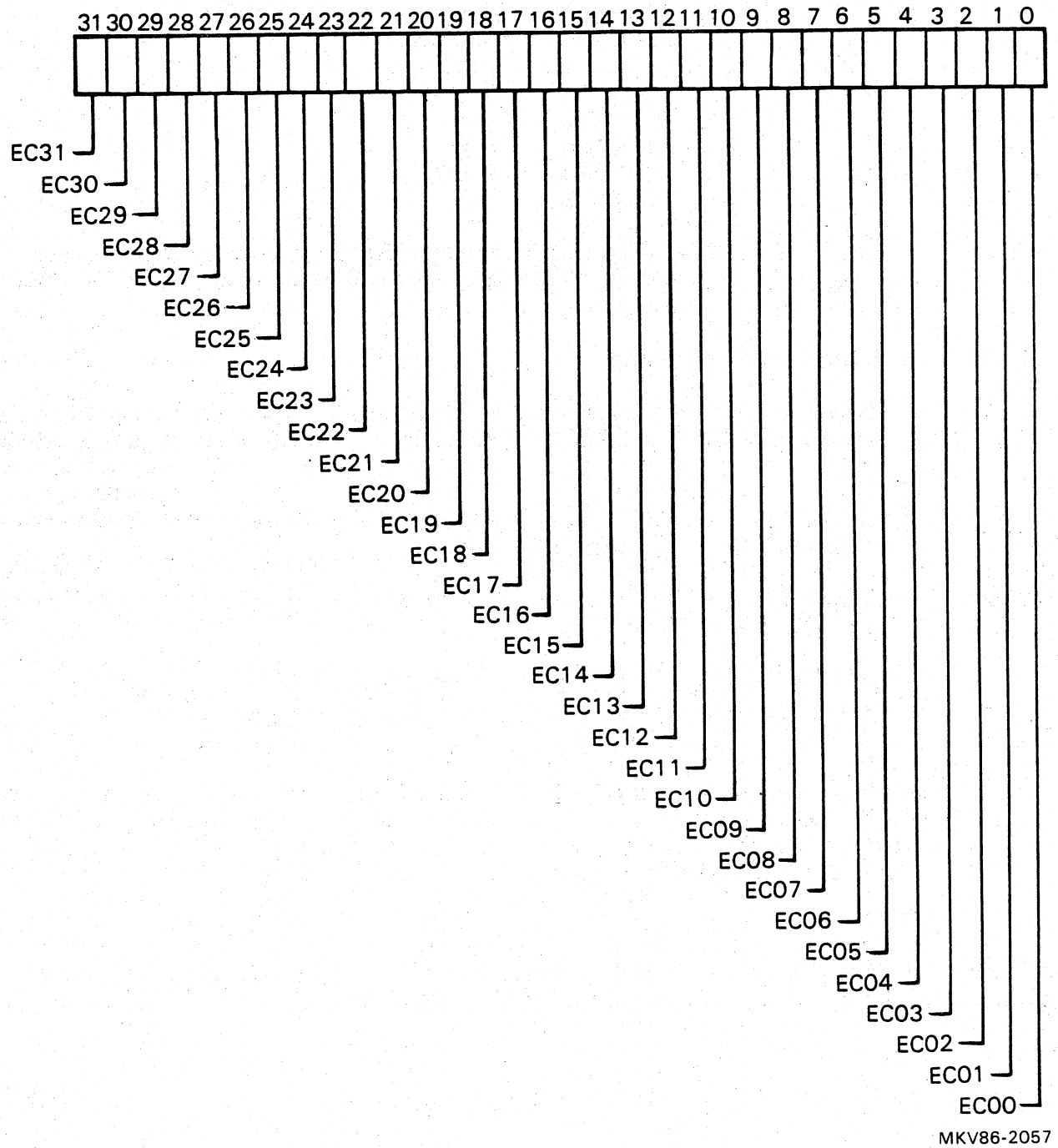


Figure A-23 Port Error Status Register

A.4.18 Port Status Register (PSR) OFFSET = 1000

The PSR returns status to the port driver after an interrupt. When an interrupt is requested by the port, these bits are fixed and are not changed until the port driver releases the register by writing the PSRCR with a "1".

The PSR is READ ONLY by the port driver and is valid only after an interrupt and before writing the PSRCR with a "1". A WRITE operation to the PSR can cause false interrupt indications to the port driver. Figure A-24 illustrates the register format. The bit assignments are described in Table A-9.

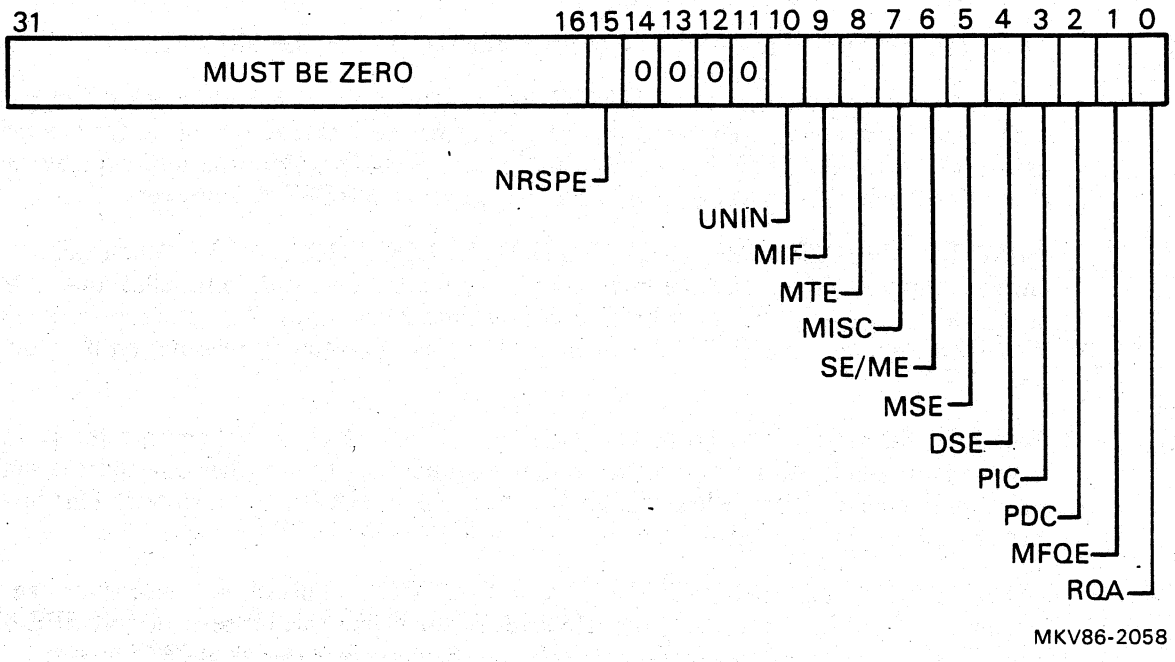


Figure A-24 Port Status Register

Table A-9 Port Status Register Bit Definitions

Bit	Description
<31:16>	Must be zero.
<15>	NO RESPONSE ERROR (DCLOC, STAC, RO) – When this bit is set, indicates that one or more of the following in the PSR is set: UNIN, MTE, MISC, SE, MSE, DSE, PIC, PDC, MFQE.
<14:11>	Reserved and read as zero.
<10>	UNINITIALIZED (DCLOC, RO, [STAS]) – When this bit is set, the port is in the uninitialized state. The port does not respond to CI traffic. MTE also sets this bit. The uninitialized state is exited by writing a "1" in the PICR or by a boot timeout.
<09>	MAINTENANCE INTERRUPT FLAG (DCLOC, RE, [STAC]) – When MIF=1, an interrupt-causing condition has occurred in the port. MIF is used with MIE to allow a diagnostic program to operate the port with interrupts disabled. MIF indicates to the program that the PSR is valid. Writing this bit has no effect.

Table A-9 Port Status Register Bit Definitions (Cont)

Bit	Description
<08>	MAINTENANCE ERROR (DCLOC, STAC, RO) – When this bit is set, the port has detected an internal hardware failure. The exact error can be determined by reading the PMCSR. When MTE is set, the port enters the uninitialized state. The microcode is halted and the port is no longer functional (except for BI slave transactions). An interrupt is generated and hardware error flags remain valid. MTE can be set as a result of any parity error flags set in the PMCSR. An MTE can occur at any time. If MTE occurs after the port has interrupted from another flag, a new interrupt is generated by the port after the PSRCR is written for the previous interrupt. After the MTE is serviced, writing the PSRCR clears the interrupt but not the error flags that caused MTE. When this bit sets, the NRSPE bit also sets.
<07>	MISCELLANEOUS ERROR DETECTED (DCLOC, STAC, RO) – When this bit is set, informs the port driver that the microcode has detected one of the miscellaneous errors and is about to enter the disabled or disabled/maintenance state. An interrupt is generated. When this bit sets, the PSR__NRSPE bit also sets. Additional information is available in the PESR.
<06>	SANITY TIMER EXPIRATION (DCLOC, STAC, RO) – When this bit is set, the maintenance sanity timer or boot timer has expired and the port has entered the uninitialized/maintenance state. When this bit sets, the PSR__NRSPE bit also sets. This bit is also referred to as the ME bit.
<05>	MEMORY SYSTEM ERROR (DCLOC, STAC, RO) – This bit sets whenever one of the hardware error bits in the Bus Error Register is set. When this bit sets, the NRSPE bit also sets. The port is in the disabled or disabled/maintenance state when MSE is set.
<04>	DATA STRUCTURE ERROR (DCLOC, STAC, RO) – When this bit is set, the port has encountered an error in a port data structure (that is; queue entry, PQB, BDT, page table, values out of range, or MBZ bits that are not zero). The port is in the disabled or disabled/maintenance state. When this bit sets, the NRSPE bit also sets.
<03>	PORT INITIALIZATION COMPLETE (DCLOC, STAC, RO) – When this bit is set, the port has completed internal initialization. The port is in the disabled or disabled/maintenance state. The local store, virtual circuit descriptor table, and the port's internal data structures are initialized. When this bit sets, the NRSPE bit also sets.
<02>	PORT DISABLE COMPLETE (DCLOC, STAC, RO) – When this bit is set, the port is disabled. It ceases processing the command queues and does not respond to incoming CI transmissions, except maintenance class, if enabled. The port is in the disabled or disabled/maintenance state. When this bit sets, the NRSPE bit also sets.
<01>	MESSAGE FREE QUEUE EMPTY (DCLOC, STAC, RO) – When this bit is set, the port attempted to remove an entry from the message free queue and found it empty. Port processing of commands continues and, therefore, the message free queue may not be empty at the time the interrupt service routine gets control. An interrupt is posted. When this bit sets, the NRSPE bit also sets.
<00>	RESPONSE QUEUE AVAILABLE (DCLOC, STAC, RO) – When this bit is set, the port has inserted an entry on an empty response queue. An interrupt is posted.

A.4.19 Port Maintenance Control/Status Register (PMCSR) OFFSET = 1004

Figure A-25 illustrates the register format. The bit assignments are described in Table A-10.

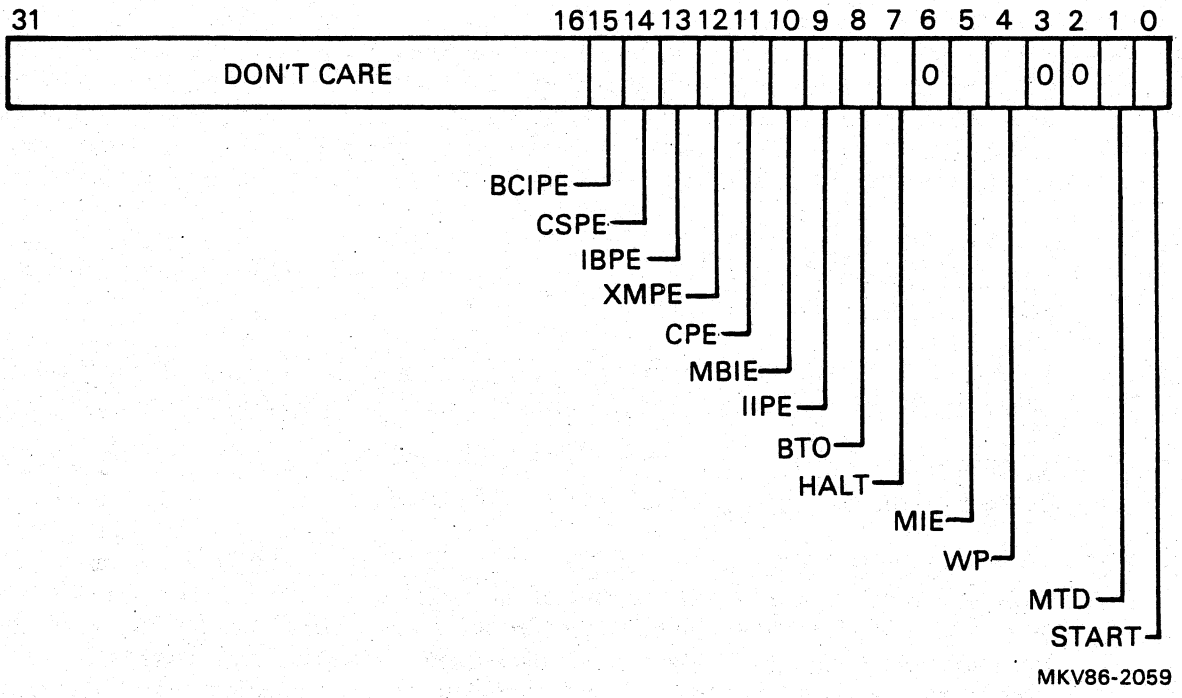


Figure A-25 Port Maintenance Control/Status Register

Table A-10 Port Maintenance Control/Status Register Bit Definitions

Bit	Description
<31:16>	Don't care.
<15>	BCI PARITY ERROR (DCLOC, STAC, RO) – This bit is set when a VAXBI or a BCI parity error occurs during a master transaction initiated by the CIBCA. When this bit is set, the Port Status Register MTE bit also sets.
<14>	CONTROL STORE PARITY ERROR (DCLOC, STAC, RO) – This bit sets when a parity error is detected in the control store RAM. CSPE can only be set when the microcode is executing. CSPE is inhibited from setting when the control store is READ in the uninitialized state (that is, a valid console or macrocode access). The MTE bit in the Port Status Register is set when this bit is set.
<13>	INTERNAL BUS PARITY ERROR (DCLOC, STAC, RO) – This bit sets when a parity error is detected on the internal bus on unsolicited WRITEs to an IB destination (including control store or on READs when the local store or VCDT is the IB source). The READs apply to both a microcode specified READ as well as to an unsolicited READ. The MTE bit in the Port Status Register is set when this bit is set.
<12>	TRANSMIT BUFFER PARITY ERROR (DCLOC, STAC, RO) – When this bit is set, indicates that a parity error was detected while the link was unloading a transmit buffer. The current CI transmission is aborted. The circuit that detects XMPE is located on the link module. The MTE bit in the Port Status Register is set when this bit is set.

Table A-10 Port Maintenance Control/Status Register Bit Definitions (Cont)

Bit	Description
<11>	CILP PARITY ERROR (DCLOC, STAC, RO) – This bit is set when a parity error is detected on the CILP bus or when a parity mismatch occurs on the IB data when the source is the CILP bus. A CPE results in the setting of the PSR__MTE via microcode. The assertion of the PSR__MTE via microcode allows the relevant status to be saved.
<10>	MAP BCI/BI ERROR (DCLOC, STAC, RO) – This bit is set when the BIIC EV error code is detected while the CIBCA is doing a MAP transaction. When this bit is set, the Port Status Register MTE bit also sets.
<09>	II PARITY ERROR (DCLOC, STAC, RO) – This bit is set when a parity error is detected on a READ over the II bus. When this bit is set, the Port Status Register MTE bit also sets.
<08>	BI BUS TIMEOUT (DCLOC, STAC, RO) – This bit is set when BTO L is asserted. It can be used by the microcode to recover from a bus timeout.
<07>	HALT SEQUENCER (R/W, DCLOC, STOPS, STAC) – When this bit is set, the sequencer halts which allows the loading of the control store RAM. In the case of the STOP command, HALT prevents the port from initiating any BI traffic. When this bit is cleared, the sequencer continues unless a power-up sequence is in progress. If HALT is set as a consequence of the BI STOP command, the result of a continuation is unpredictable. The actual loading sequence should be: a. Set the HALT bit b. Load the microcode c. WRITE the START bit, which causes the microprocessor to start running.
<06>	Reserved and read as zero.
<05>	MAINTENANCE INTERRUPT ENABLE (R/W, DCLOC, STAC) – When this bit is set, enables interrupts.
<04>	WRONG PARITY (R/W, DCLOC, STAC) – When this bit is set, the parity checker/generator on the internal bus of the port controller checks/generates even rather than odd parity. When this bit is set, the Port Status Register MTE does not halt the microsequencer.
<03:02>	Reserved and read as zero.
<01>	MAINTENANCE TIMER DISABLE (R/W, DCLOC, STAC) – When MTD=1, the boot and maintenance sanity timers are disabled and cannot cause an interrupt. When MTD=0, the timers are enabled and the PMTCR must be periodically written by the port driver to prevent the port from entering the uninitialized/maintenance state and generating an SE interrupt.
<00>	START (SC) – When this bit is set, an initialize signal is generated that clears all port errors except for the error bits internal in the CIBCA. This leaves the port in the uninitialized state. START is WRITE ONLY and cleared on power-up and at the end of the clear pulse generated by setting START. START always reads as "0" and writing a "0" has no effect.

A.4.20 Maintenance Address Register (MADR) OFFSET = 1008

The Maintenance Address Register is utilized for reading/writing the control store or the EEPROM. To READ or WRITE a particular location in either the control store or the EEPROM, the MADR has to be loaded with that particular location.

The control store is organized as 4K by 48 bits when used by the port sequencer. When the control store is accessed from the BI, it is organized as 12K by 16 bits. When accessing the EEPROM from the BI, the EEPROM is organized as 8K by 8 bits.

The MADR is WRITE ONLY when the port is in the uninitialized state and the MADR is not cleared by START.

The MADR is incremented automatically at the end of a MDATR WRITE but not at the end of a MDATR READ.

At the end of a WRITE to the control store, the MADR increments to point to the next 16-bit section. This means that MADR address bits <12:00> are incremented only at the end of a WRITE to control store bits <47:32> and that MADR address bits <A15:A14> are incremented at the end of each WRITE to a control store section. A section of the control store can be either bits <47:32>, <31:16>, or <15:00>.

At the end of a WRITE to the EEPROM, the MADR bits <13:00> are incremented, however, MADR bits <15:14> remain unchanged.

The incrementing of the MADR at the end of each WRITE to either the control store or to the EEPROM, allows for sequential WRITEs to the MDATR without having to update the MADR. The MADR is not incremented at the end of a READ of either the control store or the EEPROM.

When the port is not in the uninitialized state, a READ returns undefined data and a WRITE has no effect. Figure A-26 illustrates the register format. The bit assignments are described in Table A-11. Figure A-27 is a map of the control strore.

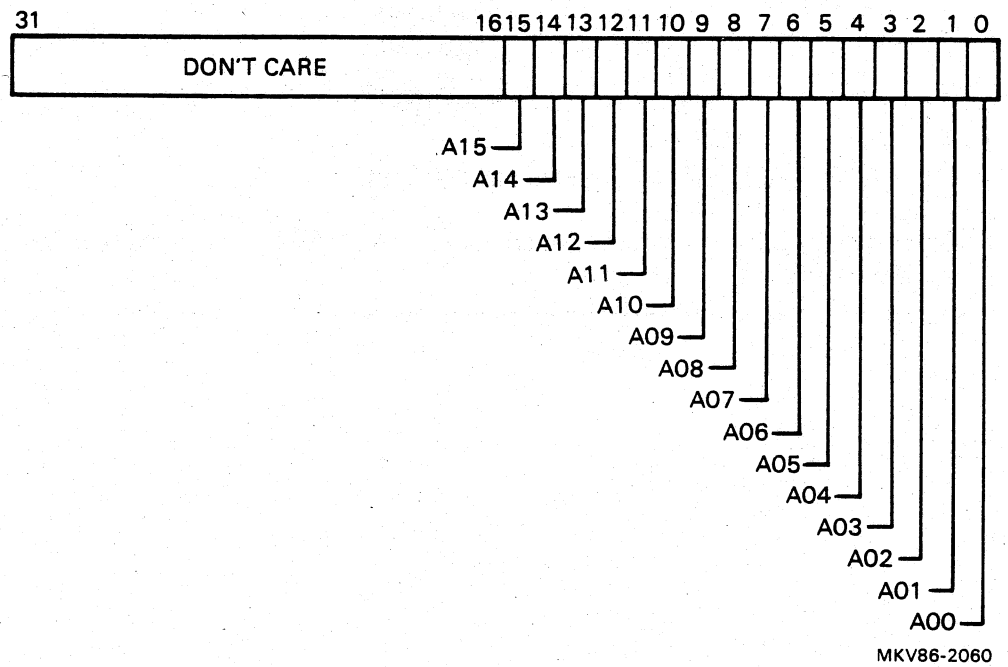


Figure A-26 Maintenance Address Register

Table A-11 Maintenance Address Register Bit Definitions

Bit	Description
<31:16>	Don't care.
<15:14>	This field selects bits of control store or EEPROM as follows:

A15	A14	Selects Bits
0	0	<15:00> of the control store
0	1	<31:16> of the control store
1	0	<47:32> of the control store
1	1	<07:00> of the EEPROM

<13>	Reserved.
<12:00>	The meaning of these bits depends on the value of address bits <A15:A14> as follows:

With address bits <A15:A14> equal to 00, 01, or 10; and <A12> equal to zero:

A11	A10	Bank Selected	Address Range
0	0	0	000–3FF
0	1	1	400–7FF
1	0	2	800–BFF
1	1	3	C00–FFF

<A09:A00> select the word within the 1K bank.

With address bits <A15:A14> equal to 11, <A12:A00> select the location (byte) within the 8K EEPROM.

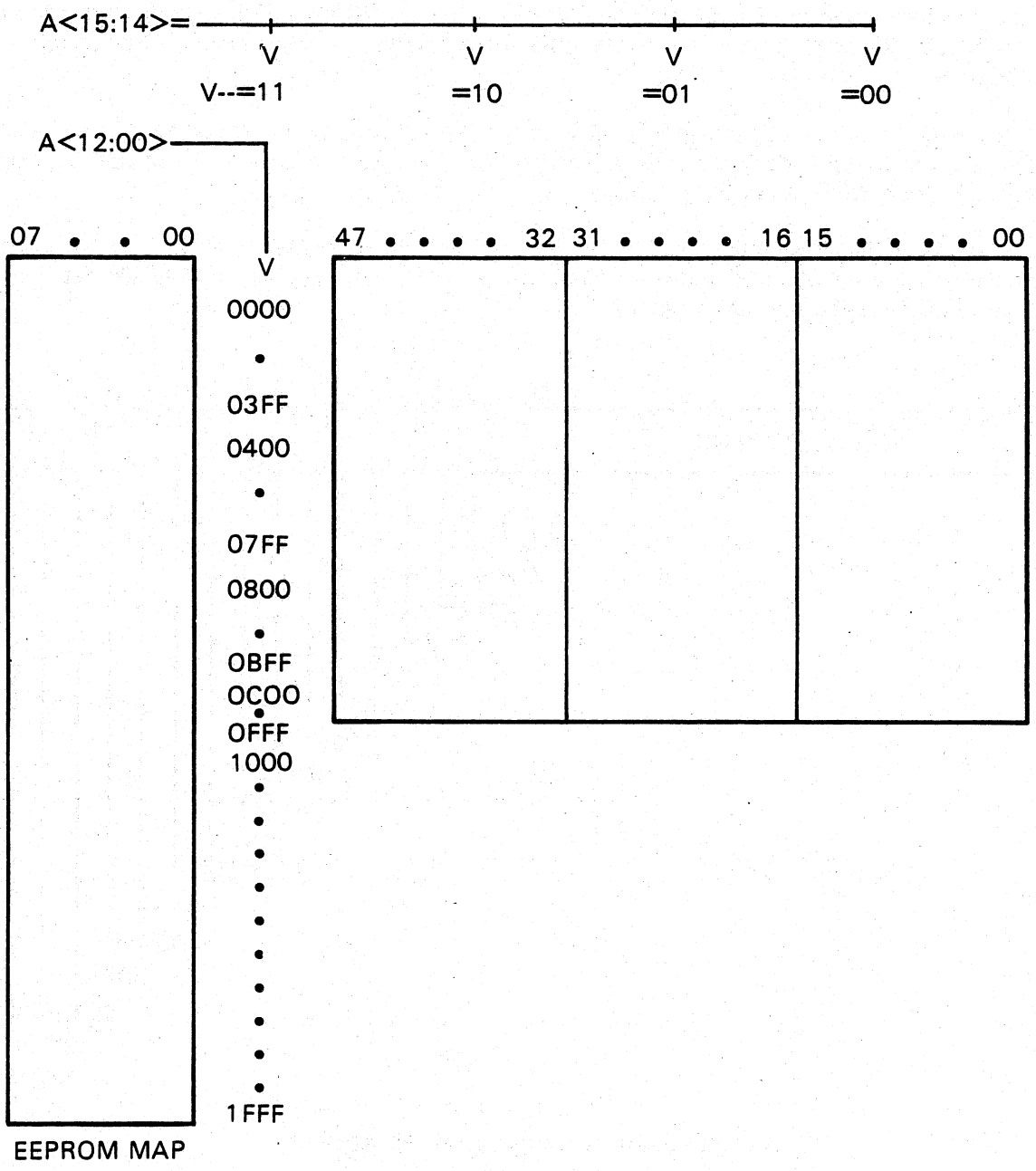


Figure A-27 Control Store Map

A.4.21 Maintenance Data Register (MDATR) (R/W) OFFSET = 100C

When writing the Maintenance Data Register, the data is written into the address specified by the MADR (which can be a location in the control store or a location in the EEPROM). The upper 16 bits <31:16> of the MDATR must be zero. At the end of the WRITE, the MADR is incremented to point to the next section/location.

When reading the MDATR, the data is READ from the address specified by the MADR (which can be a location in the control store or a location in the EEPROM). The upper 16 bits are "don't care". At the end of the READ, the MADR is not incremented.

The MDATR is only valid when the port is in the uninitialized state with HALT in the PMCSR set. When the port is not in the uninitialized state, a READ returns undefined data and a WRITE has no effect. Figure A-28 illustrates the register format.

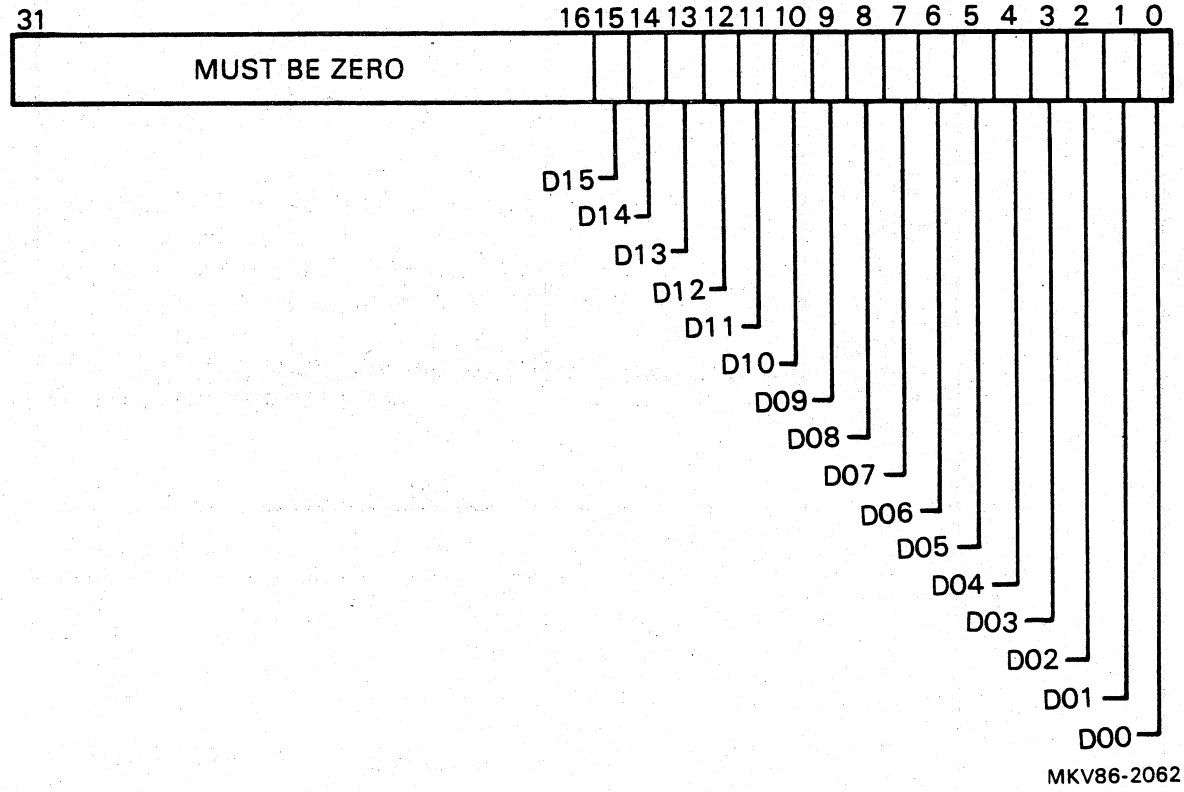


Figure A-28 Maintenance Data Register

A.4.22 Port Command Queue 0 Control Register (PCQ0CR) OFFSET = 1010

When the port driver inserts an entry in an empty Command Queue 0, the port driver WRITEs a "1" in the PCQ0CR to initiate port processing of the command queue. The PCQ0CR is ignored if the port is in the uninitialized, uninitialized/maintenance, disabled, or disabled/maintenance state.

The PCQ0CR is WRITE ONLY. Reading this register returns undefined data. Writing a "0" has no effect. Figure A-29 illustrates the register format.

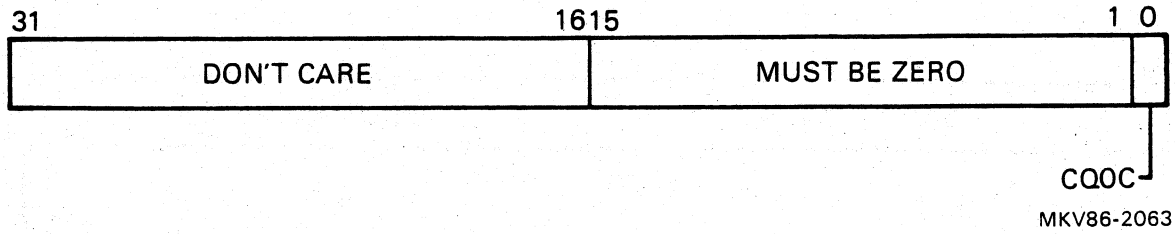


Figure A-29 Port Command Queue 0 Control Register

A.4.23 Port Command Queue 1 Control Register (PCQ1CR) OFFSET = 1014

When the port driver inserts an entry in an empty Command Queue 1, the port driver WRITEs a "1" in the PCQ1CR to initiate port processing of the command queue. The PCQ1CR is ignored if the port is in the uninitialized, uninitialized/maintenance, disabled, or disabled/maintenance state.

The PCQ1CR is WRITE ONLY. Reading this register returns undefined data. Writing a "0" has no effect. Figure A-30 illustrates the register format.

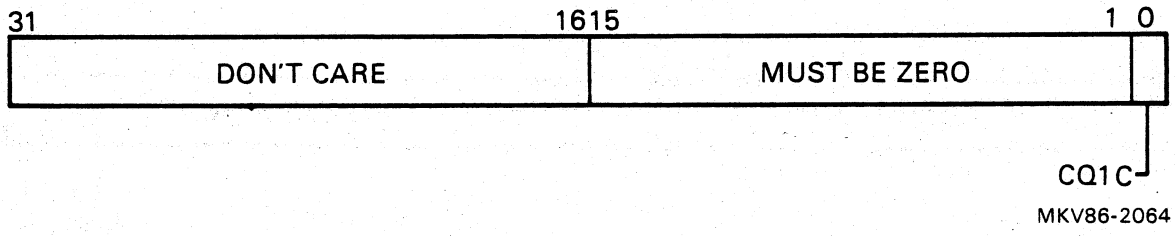


Figure A-30 Port Command Queue 1 Control Register

A.4.24 Port Command Queue 2 Control Register (PCQ2CR) OFFSET = 1018

When the port driver inserts an entry in an empty Command Queue 2, the port driver WRITEs a "1" in the PCQ2CR to initiate port processing of the command queue. The PCQ2CR is ignored if the port is in the uninitialized, uninitialized/maintenance, disabled, or disabled/maintenance state.

The PCQ2CR is WRITE ONLY. Reading this register returns undefined data. Writing a "0" has no effect. Figure A-31 illustrates the register format.

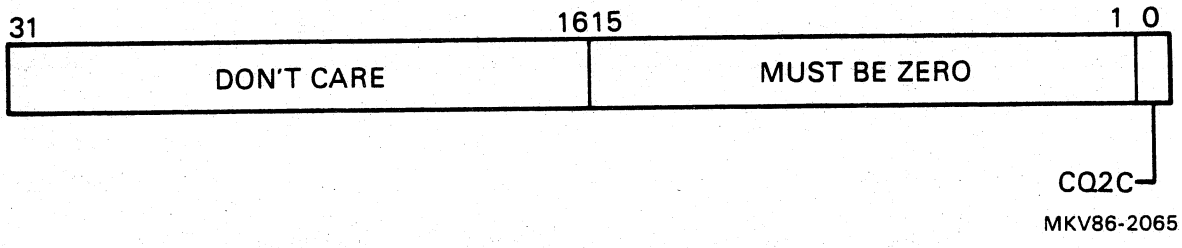


Figure A-31 Port Command Queue 2 Control Register

A.4.25 Port Command Queue 3 Control Register (PCQ3CR) OFFSET = 101C

When the port driver inserts an entry in an empty Command Queue 3, the port driver WRITEs a "1" in the PCQ3CR to initiate port processing of the command queue. The PCQ3CR is ignored if the port is in the uninitialized, uninitialized/maintenance, disabled, or disabled/maintenance state.

The PCQ3CR is WRITE ONLY. Reading this register returns undefined data. Writing a "0" has no effect. Figure A-32 illustrates the register format.

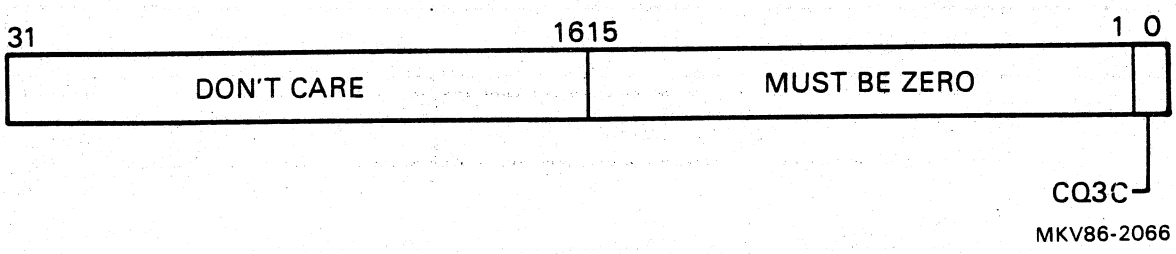


Figure A-32 Port Command Queue 3 Control Register

A.4.26 Port Status Release Control Register (PSRCR) (SC,VMSL) OFFSET = 1020

After the port driver has received an interrupt and READ the PSR, it returns the PSR to the port by writing a "1" in the PSRCR. The PSR is invlaid until the next interrupt is received from the port. The PSRCR is ignored if the port is in the uninitialized, uninitialized/maintenance, disabled, or disabled/maintenance state.

The PSRCR is WRITE ONLY. Reading this register returns undefined data. Writing a "0" into this register has no effect. Figure A-33 illustrates the register format.

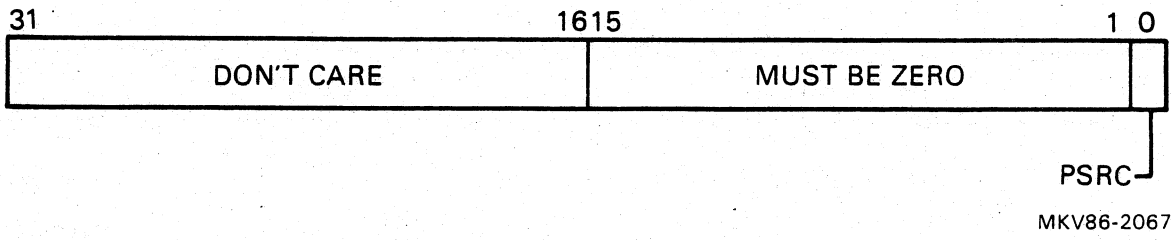


Figure A-33 Port Status Release Control Register

A.4.27 Port Enable Control Register (PECR) (SC,VMSL) OFFSET = 1024

The port driver enables the port by writing a "1" in the PECR. The PECR is WRITE ONLY and the WRITE is ignored if the port is in the uninitialized, uninitialized/maintenance, enabled, or enabled/maintenance state. Reading the PECR returns undefined data. Writing a "0" into this register has no effect. Figure A-34 illustrates the register format.

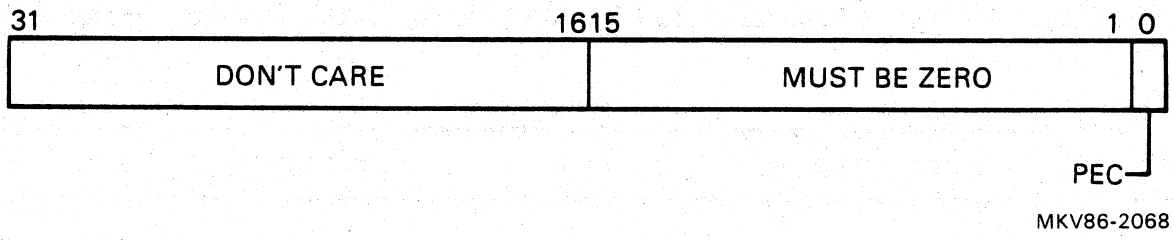


Figure A-34 Port Enable Control Register

A.4.28 Port Disable Control Register (PDCR) (SC,VMSL) OFFSET = 1028

The port driver enables the port by writing "1" in the PDCR. When the port is disabled, it requests an interrupt with the PDC bit in the PSR set.

The PDCR is WRITE ONLY and is ignored if the port is in the uninitialized, uninitialized/maintenance, disabled, or disabled/maintenance state. Reading the PDCR returns undefined data. Writing a "0" into this register has no effect. Figure A-35 illustrates the register format.

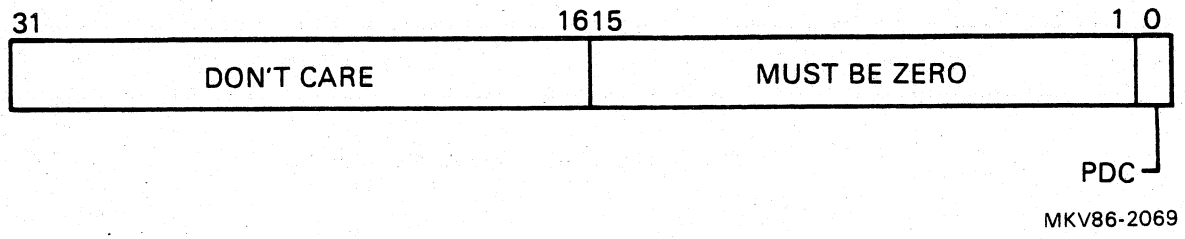


Figure A-35 Port Disable Control Register

A.4.29 Port Initialize Control Register (PICR) (SC,VMSL) OFFSET = 102C

The port driver initializes the port by writing a "1" in the PICR. When the initialization is complete, the port sets the PIC bit in the PSR and requests an interrupt. The port enters the disabled state.

The PICR is WRITE ONLY and is ignored if the port is in the disabled or disabled/maintenance state. If the PICR is written with a "1" while the port is in the enabled or enabled/maintenance state, the port will go to the disabled or disabled/maintenance state with loss of processing state. Reading this register returns undefined data. Writing a "0" into this register has no effect. Figure A-36 illustrates the register format.

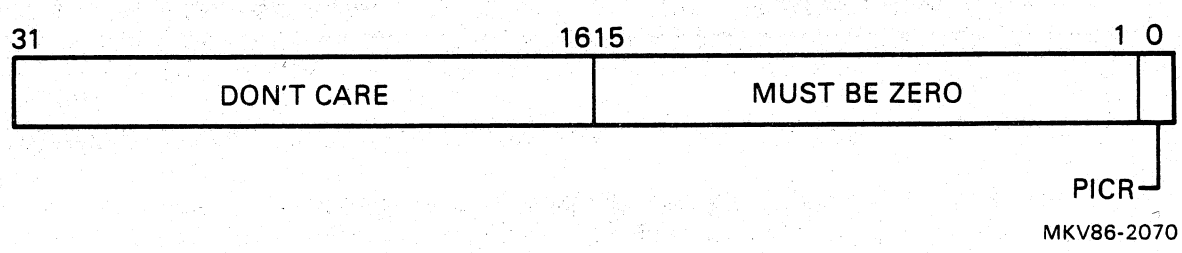


Figure A-36 Port Initialize Control Register

A.4.30 Port Datagram Free Queue Control Register (PDFQCR) (SC,VMSL) OFFSET = 1030

The PDFQCR is written with a "1" by the port driver when the datagram free queue is found to be empty at the time of a datagram free queue entry insertion.

The PDFQCR is WRITE ONLY and is ignored if the port is in the uninitialized, uninitialized/maintenance, disabled, or disabled/maintenance state. Writing a "0" into this register has no effect. Figure A-37 illustrates the register format.

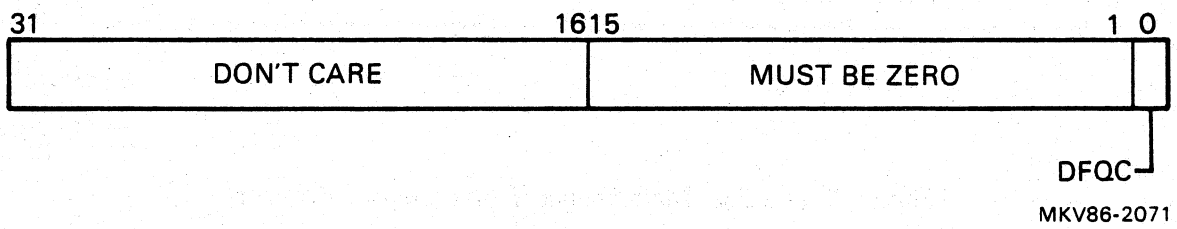


Figure A-37 Port Datagram Free Queue Control Register

A.4.31 Port Message Free Queue Control Register (PMFQCR) (SC,VMSL) OFFSET = 1034

The PMFQCR is written with a "1" by the port driver when the message free queue is found to be empty at the time of a free queue 1 entry insertion. If the message free queue is exhausted and the port has posted an MFQE interrupt, the port will wait until the PMFQCR has been written before it removes a free queue entry.

The PMFQCR is WRITE ONLY and is ignored if the port is in the uninitialized, uninitialized/maintenance, disabled, or disabled/maintenance state. Writing a "0" into this register has no effect. Figure A-38 illustrates the register format.

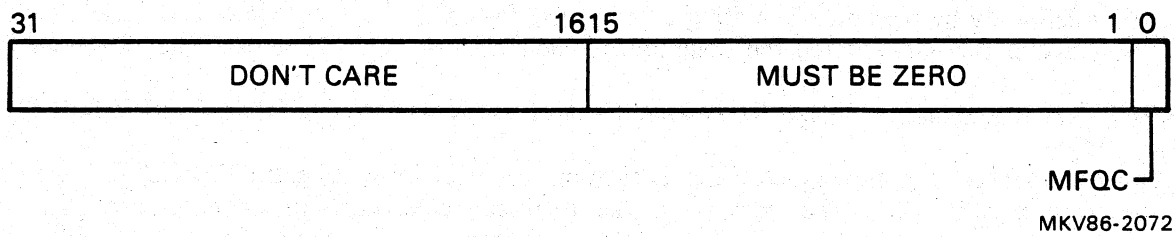


Figure A-38 Port Message Free Queue Control Register

A.4.32 Port Maintenance Timer Control Register (PMTCR) (SC,VMSL) OFFSET = 0038

The PMTCR allows the macrocode to control the expiration times of the boot and sanity timers. The timers are reset to their initial values when a "1" is written into the PMTCR. The PMTCR is WRITE ONLY. Reading this register returns undefined data. Writing a "0" into this register has no effect. Figure A-39 illustrates the register format.

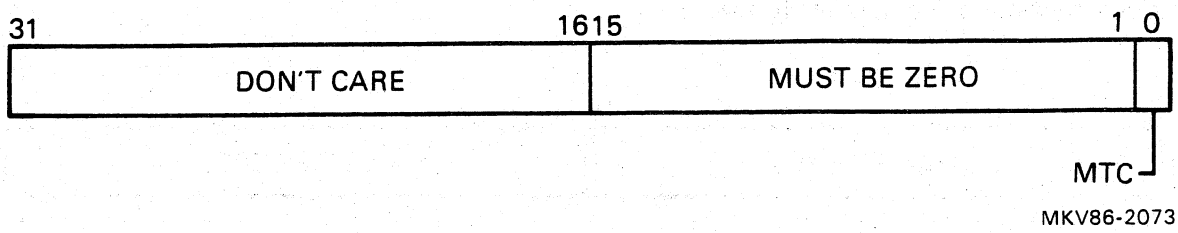


Figure A-39 Port Maintenance Timer Control Register

A.4.33 Sanity Timer

The sanity timer is implemented in microcode by branching on a hardware time base. It is reset when the PMTCR is written with a "1". If the macrocode fails to WRITE a "1" in the PMTCR within 100 seconds, the port will post an interrupt and enter the uninitialized/maintenance state with SE=1 in the Port Status Register. The sanity timer is disabled if MTD=1 in the PMCSR.

A.4.34 Boot Timer

The delay time is selected by backplane setup.

If START is set and the PMCTR is written with a "1", the uninitialized state will be exited after the preset delay unless the time was set for "0" seconds. If the jumpers were set for "0" delay, the port will wait 50 seconds before exiting the uninitialized state. If MTE=1 or MTD=1 the microcode will not start.

After the boot timer expires, the uninitialized bit in the PSR is cleared. The port then posts an interrupt and enters the uninitialized/maintenance state with SE=1 in the PSR. The expiration time can be extended indefinitely by periodically writing a "1" in the PMTCR. The boot timer is disabled if MTD=1 in the PMCSR or the port entered the uninitialized state because MTE=1 in the Port Status Register.

A.4.35 Port Maintenance Timer Expiration Control Register (PMTECR) (SC,VMSL) OFFSET = 103C

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, or disabled/maintenance state and only while the maintenance timer is not disabled. Figure A-40 illustrates the register format.

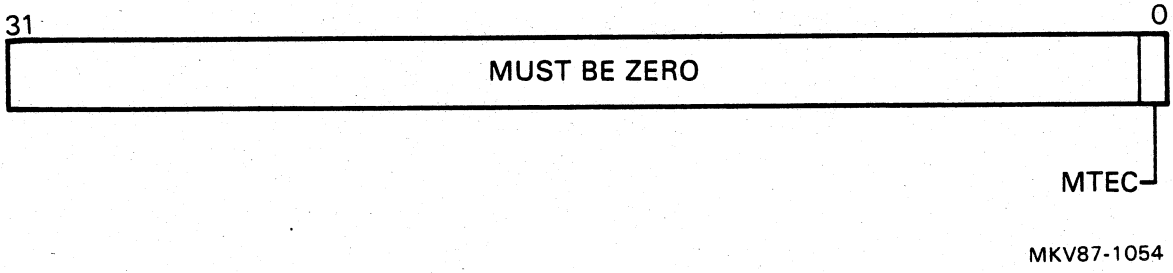


Figure A-40 Port Maintenance Timer Expiration Control Register

Digital Equipment Corporation • Bedford, MA 01730