
DRB32

DRB32 Hardware Installation Guide

Order Number: EK-DRB32-IN-002

This book describes the installation of the DRB32 adapter. It also provides information on running the DRB32 diagnostics, troubleshooting, and installing option modules.

digital equipment corporation
maynard, massachusetts

January 1987
July 1990

The DRB32 is a high-speed parallel port sold by Digital Equipment Corporation for use in qualified applications. Contact your Digital sales representative for more information.

The information in this document is subject to change without notice and should not be construed as a commitment by Digital Equipment Corporation. Digital Equipment Corporation assumes no responsibility for any errors that may appear in this document.

The software described in this document is furnished under a license and may be used or copied only in accordance with the terms of such license.

No responsibility is assumed for the use or reliability of software on equipment that is not supplied by Digital Equipment Corporation or its affiliated companies.

Digital Equipment Corporation makes no representation that the interconnection of its products in the manner described herein will not infringe on existing or future patent rights, nor do the descriptions contained herein imply the granting of license to make, use, or sell equipment constructed in accordance with this description.

The following are trademarks of Digital Equipment Corporation:

DEC	PDP	VAXcluster
DECnet	ULTRIX	VMS
DECUS	UNIBUS	[redacted]
DRB32	VAX	
DR11-W	VAXBI	

Contents

Figures

Tables

Preface

Purpose of this Manual

The *DRB32 Hardware Installation Guide* (EK-DRB32-IN) supplies installation information for the DRB32 adapter. It covers the hardware installation, information on how to run the diagnostics, and troubleshooting the DRB32 and option modules.

Information on installing the DRB32 software is in the *DRB32 Programmer's Manual* (AA-HZ25C-TE).

Intended Audience

This manual is for:

- Digital or customer personnel who install and repair this equipment in the field.
- Customer engineers and programmers who incorporate this equipment into their own product or system.

Structure of this Manual

Chapter 1 describes the DRB32 adapter.

Chapter 2 describes installation of the DRB32 hardware.

Chapter 3 describes testing and verification of the DRB32 hardware.

Appendix A contains DRB32 Output Tester patterns.

Appendix B contains the error log format.

Appendix C lists the event codes that can appear in the DRB32 ERREG Register.

Appendix D lists the self-test error codes.

Appendix E lists the DRB32 registers.

Appendix F is a glossary that describes the DRB32 hardware and software.

Related Manuals

The DRB32 adapter is one of a family of processors, memories, and adapters that uses the 32-bit VAXBI bus. For a technical summary of all VAXBI modules, system components, and integrated circuits, see the *VAXBI Options Handbook* (Order No. EB-27271-46).

Other related technical manuals are:

DRB32 Introduction
Order No. EK-DRB32-OV

An overview of the hardware, software, and use of the DRB32 adapter.

DRB32 Technical Manual
Order No. EK-DRB32-TM

Description of the DRB32 hardware and functional operation for the hardware designer, including register descriptions, data path descriptions, pinouts, and VAXBI transactions.

DRB32 Programmer's Manual
Order No. AA-HZ25C-TE

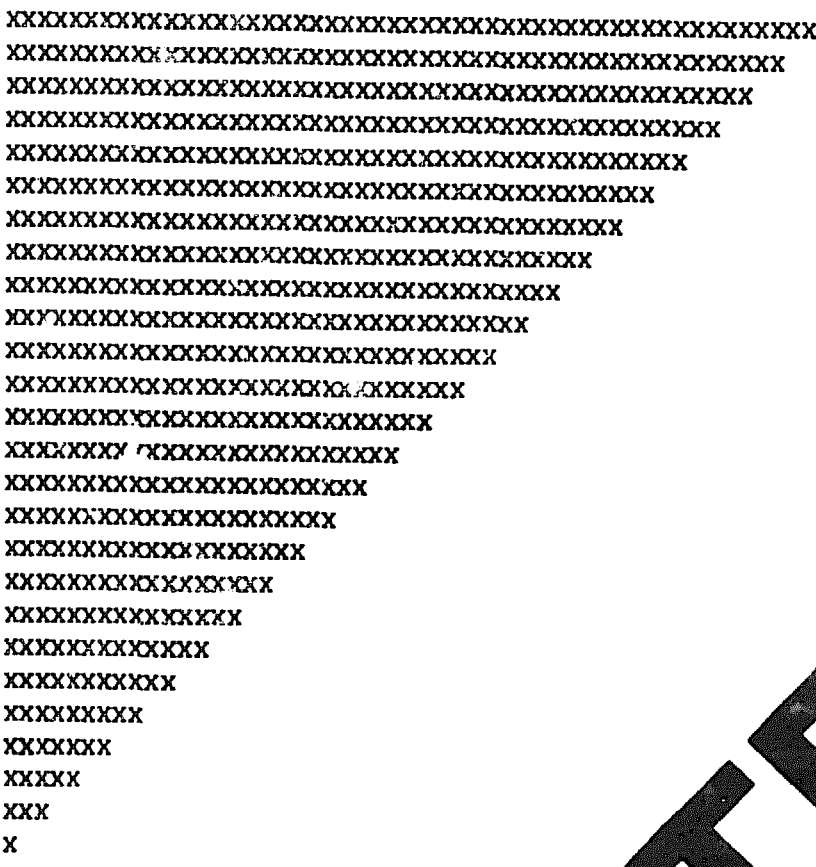
Description of the DRB32 software for the software programmer and designer, including device drivers, tools, example programs and test code. Also describes software installation procedures.

System Installation Guide

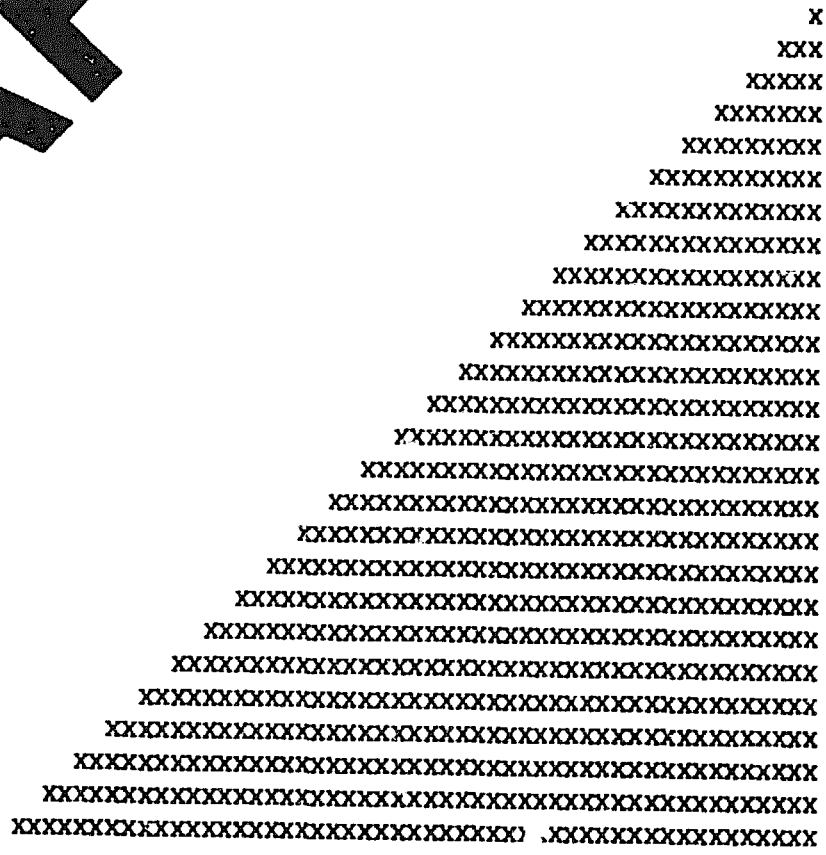
Description of the cable management practices, how to access the VAXBI card cage, and other host-specific requirements.

Conventions Used

- The DRB32 adapter is referred to as the DRB32. "DRB32" refers to the DRB32-M option, if no option is specified.
- The VAXBI bus is referred to as the VAXBI.



CHAPTER 1



1 Introduction

The *DRB32 Hardware Installation Guide* describes the installation of the DRB32 hardware. This manual also includes information on DRB32 diagnostics, troubleshooting, and installing option modules. See the *DRB32 Programmer's Guide* for information on installing the DRB32 software.

1.1 Functional Description of the DRB32 Module

The DRB32 is an adapter that provides fast data transfers between Digital's VAXBI systems and user devices such as array processors, A/D converters, and high-speed instrumentation. The DRB32 meets the requirements of a VAXBI node, and therefore provides a simple connection for your devices to VAXBI systems. Large, fast data transfers are the major advantages of the DRB32. The DRB32 is optimized for large DMA transfers in **Block mode**.

Block mode allows for transfers up to 982,528 bytes without processor intervention. The DRB32 hardware has a double buffering capability that enables it to set up DMA transfers that are larger than this 982,528-byte limit. Data transfers can also be made in Data mode, in which a host processor on the VAXBI directly reads or writes the DRB32 register that sends and receives data from the user device.

The DRB32 uses map registers to translate virtual addresses to VAXBI physical addresses. The DRB32 map register area is 1919 longwords. Mapped adapters require a process running in a VAX computer to load map registers into the adapter for virtual-to-physical address conversion. Transfer widths are selected by the user—the DRB32 can handle longword (32-bit), word (16-bit), or byte (8-bit) transfers. However, the selection of 16- or 8-bit widths results in lower performance. The DRB32 uses odd parity.

The major functional components of the DRB32 are the following:

- Parallel I/O port
- VAXBI interface
- T-11 subsystem

The DRB32 is a parallel I/O port (with a 32-bit half-duplex data path and dual 8-bit duplex control paths) that connects the VAXBI bus and a user device. At the VAXBI end, the DRB32 has a VAXBI interface corner. At the user device end, the DRB32 parallel port has a user interface with signals for data, control, and device status. Attached to the parallel I/O port is a T-11 subsystem that performs module self-test. A simplified block diagram of these three DRB32 functional components is shown in Figure 1-1 below.

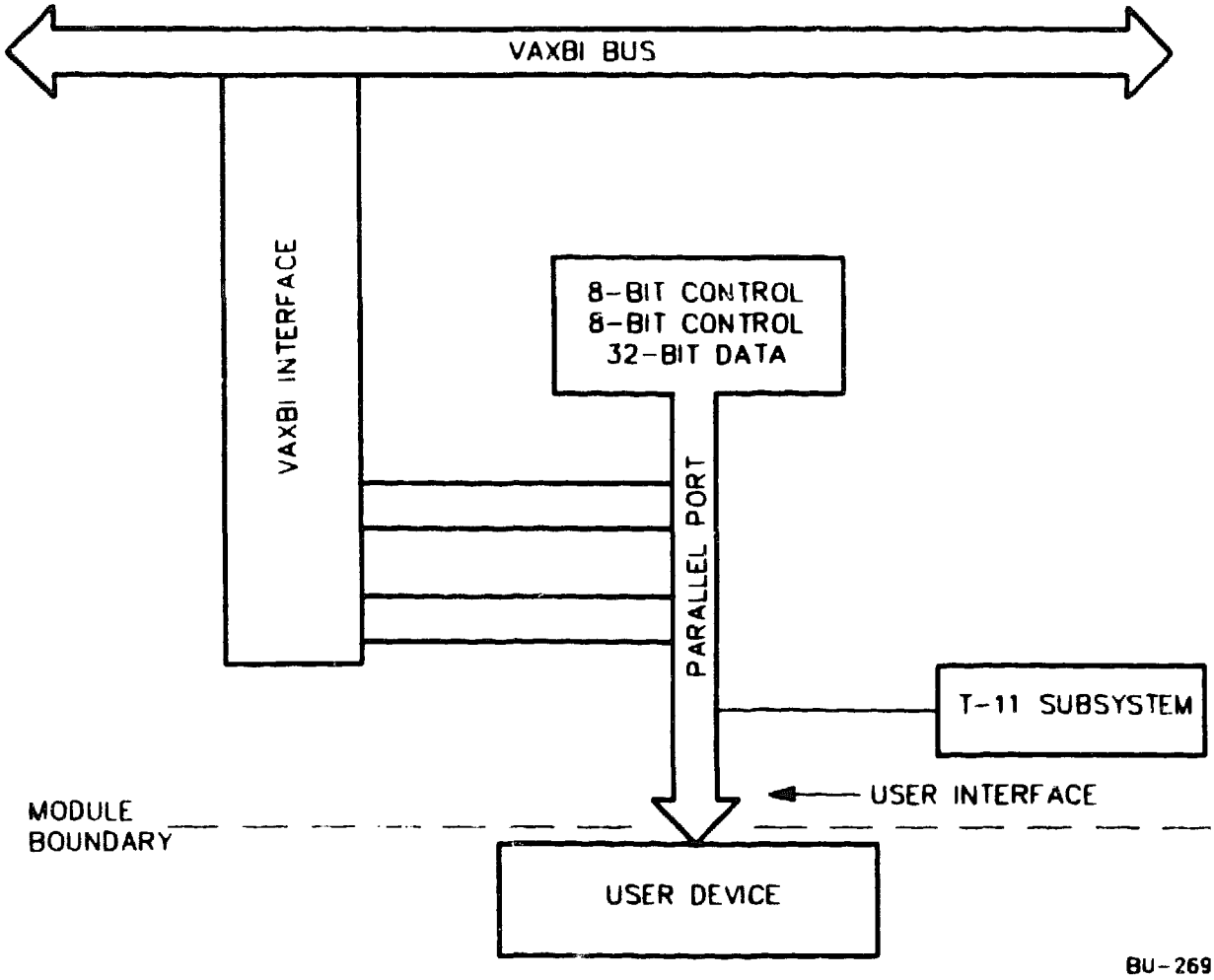


Figure 1-1 DRB32 Simplified Block Diagram

1.2 DRB32 Configurations

The DRB32 is available in three configurations:

- DRB32-M
- DRB32-E
- DRB32-W

1.2.1 DRB32-M Option

The DRB32-M is the basic intracabinet DRB32 option, designed for an application where all cabling is kept within a single cabinet. This option consists of the primary DRB32 module (T1022) and a set of cables. The DRB32-M is compliant with FCC Class A regulations when fully contained within an FCC-compliant cabinet. For a more detailed description of this option, see Section 2.2.1.

1.2.2 DRB32-E Option

The DRB32-E external cable-driving option can drive an intercabinet cable that is up to 40 feet long. The DRB32-E is for applications in which user equipment must be separate from the cabinet containing the DRB32. Therefore, a cable must connect the two cabinets. This option consists of the primary DRB32 module (T1022), the external driver module (T1024), an I/O connector panel, and cables. For a more detailed description of this option, see Section 2.2.2.

1.2.3 DRB32-W Option

The DRB32-W option provides a means of connecting DR11-W equipment to VAXBI systems. The DRB32-W is for intracabinet use. It consists of the primary DRB32 module (T1022), a DR11-W converter module (T1023), a bulkhead connector, and cables. For a more detailed description of options, see Section 2.2. For a more detailed description of this option, see Section 2.2.3.

1.3 DRB32 Software

The DRB32 software is included in a VMS layered product called DRB32VMSDRIVERS and must be ordered separately.

The following software is available for the DRB32-M and DRB32-E:

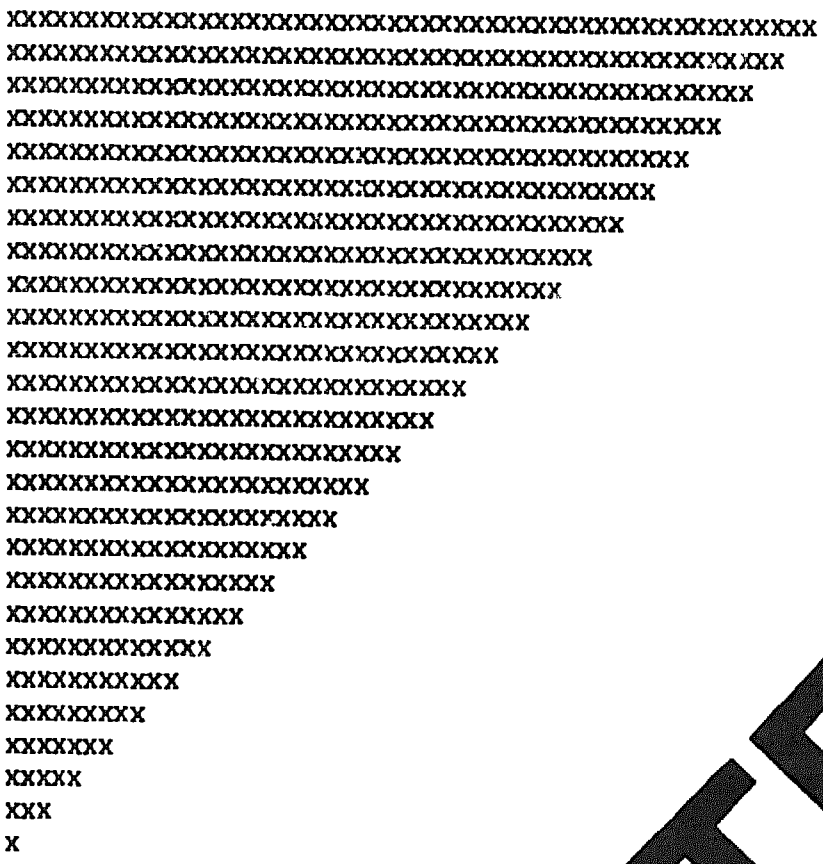
- VMS device driver (UQDRIVER) for the DRB32-M/-E
- Test code for drivers (DRB32$QIO)
- Subroutine package for communications between two VAXBI systems through a back-to-back DRB32 link (DRB32$MESSAGE)
- MACRO and object libraries
- Command procedures to load and build the driver and test programs

The following software is available for the DRB32-W:

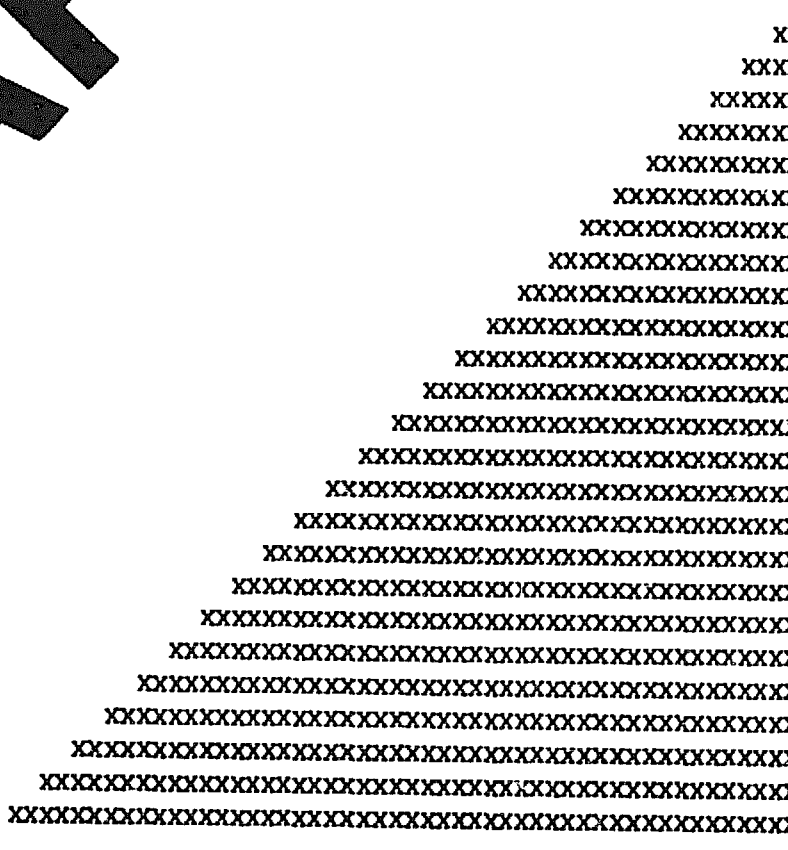
- VMS device driver (UQWDRIVER) for the DRB32-W
- Test code for drivers (DRB32$WQIO)
- MACRO and object libraries
- Command procedures to load and build the driver and test programs

The software listed above is described in the *DRB32 Programmer's Manual*, as are the instructions for software installation and verification.

Hardware diagnostics are available and are described in Chapter 3 of this manual.



CHAPTER 2



2 DRB32 Hardware Installation Procedures

This chapter describes the hardware installation of the DRB32-M, the DRB32-E, and the DRB32-W.

CAUTION
You must wear an antistatic wrist ground strap 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.

2.1 Summary of Installation Tasks

The following steps are necessary for successful installation of the DRB32 options.

1. Unpack and inspect the Installation Kit.
2. Check VAXBI configuration.
3. Install the DRB32 module(s).
4. Install internal cables.
5. Install, if necessary, transition header(s), I/O connector panel, and/or bulkhead connector.
6. Install, if necessary, any external cables.
7. Test and verify the installation.

2.1.1 Site Preparation

There is no special site preparation necessary for the installation of any of the DRB32 options. To install the DRB32, there must be space in the VAXBI card cage. If you are installing a DRB32-E or DRB32-W, two contiguous VAXBI slots in the same cage are needed.

WARNING
It is essential that all devices be connected to the same line-power bus to ensure that all devices have the same chassis ground. A danger to personnel and a fire hazard exist when various devices have different chassis ground potentials.

2.1.2 DRB32 Option Components

The DRB32 and any associated option modules, I/O connector panels, and bulkhead connectors are supplied in the installation kits. Use Table 2-1 to check the contents of the installation kit. Examine each part for damage and make sure all parts are present. Report any damage to the shipper and to the local Digital office. Requisition any missing parts.

Three installation kits are available for the three DRB32 options. The contents of these three kits are described in Table 2-1.

Table 2-1 Contents of DRB32 Options

Part Number	Description	DRB32-M Option	DRB32-E Option	DRB32-W Option
T1022-AA	DRB32-M Module	1	1	1
T1024-AA	DRB32-E External Driver Module	0	1	0
T1023-YA	DRB32-W Converter Module	0	0	1
17-01474-01	Intermodule Cable	0	3	3
70-23924-01	I/O Connector Panel	0	1	0
70-23923-01	Bulkhead Connector	0	0	1
EK-DRB32-IN	DRB32 Hardware Installation Guide	1	1	1

Cabinet kits, internal cables, and external cables are not part of the installation kit and are ordered separately. The customer orders any of the following cables for installation of their equipment. Each part number represents a set of three cables. For the DRB32-M and for the DRB32-W, there must be one set of internal cables. For the DRB32-E, there must be a set of internal cables and a set of external cables. See Table 2-2.

Table 2-2 Cable Ordering Numbers

Option Number	Description	DEC Part Number
CK-DRB32-LJ	Set of Three Internal Cables 5-foot Length	17-01375-02
CK-DRB32-LM	Set of Three Internal Cables 10-foot Length	17-01375-03
CK-DRB32-LN	Set of Three Internal Cables 15-foot Length	17-01375-04
BS17Y-20	Set of Three BC17Y-20 External Cables, 20 feet	17-00445-06
BS17Y-40	Set of Three BC17Y-40 External Cables, 40 feet	17-00445-09

2.1.3 Configuration Rules

The DRB32 address space is determined by the NODE plug associated with the slot where the module is located. You do not have to set any jumpers or special addresses to install the hardware. There are, however, some VAXBI rules that must be followed. These rules apply in all VAXBI systems.

1. No DRB32 module can be placed in slot J1K1 (the rightmost slot).
2. The DRB32-E or DRB32-W requires two adjacent VAXBI slots in the same cage. The DRB32-M module (T1022) goes in the rightmost of the two slots.
3. The VAXBI node ID for the DRB32 must be unique. The node ID is found on a plug on the backplane, opposite to the slot containing the DRB32-M module. VAXBI systems are shipped with node plugs already installed.

There may be other system-dependent configuration restrictions, such as those imposed by backplane capacity, or physical mounting limitations in the cabinet. Such restrictions are described in the host system documentation.

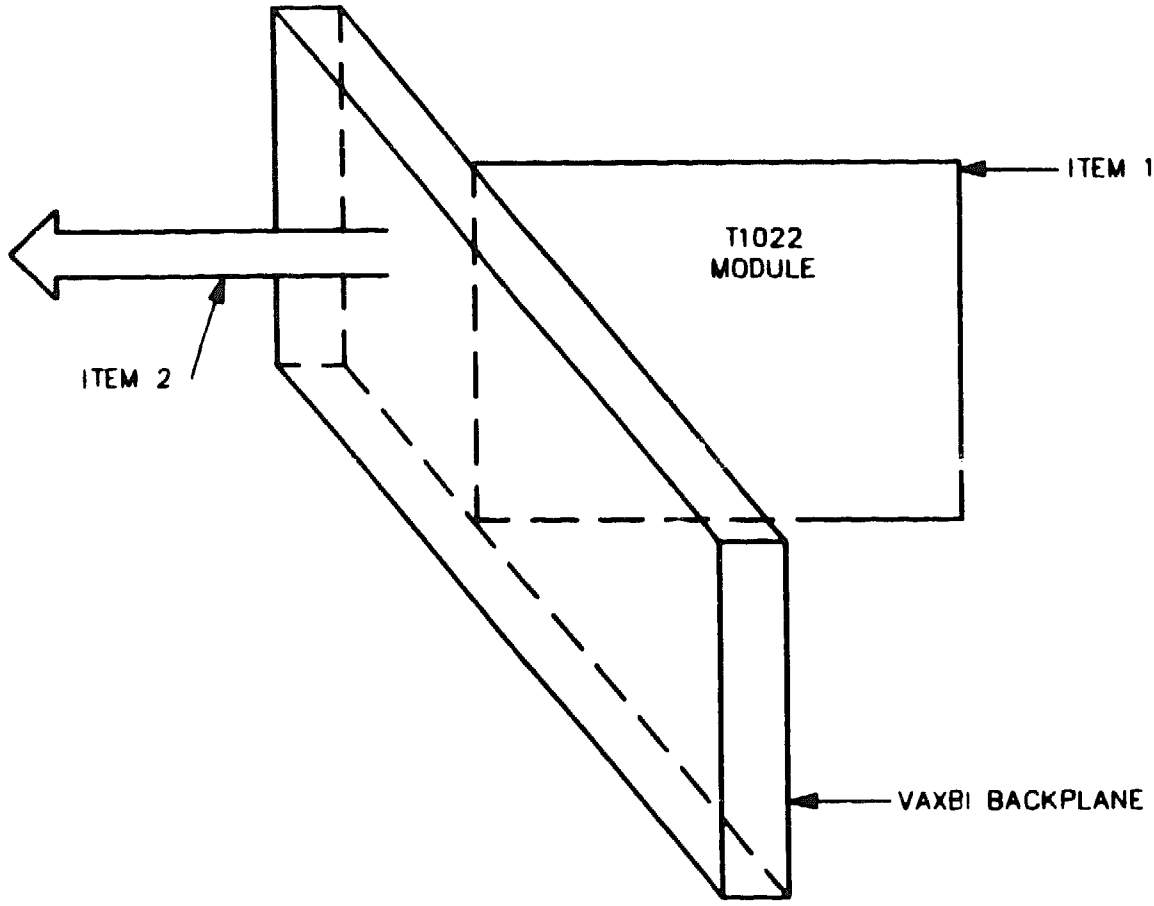
2.2 Option Installation

WARNING
Remove all power to the system before performing the module installation.

CAUTION
You must wear an antistatic wrist ground strap 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.

2.2.1 DRB32-M Installation

Figure 2-1 shows how the DRB32-M fits together mechanically. Use the figure and the following instructions to install the option.



ORDER NO.	ITEM	QUANTITY	DESCRIPTION
DRB32-M	1	1	BASE MODULE (T1022)
CK-DRB32-L*	2	1 SET (3)	INTERNAL CABLES (UP TO 15 FT)

BU-2691

Figure 2-1 DRB32-M Module

Follow the steps below for installing the DRB32-M.

1. Expose the back of the VAXBI card cage.

2. Insert the DRB32 (T1022) module into an empty VAXBI slot, following the configuration rules in section 2.1.3. All VAXBI modules are keyed to prevent incorrect installation.
3. Ensure that a unique node ID plug is present on the VAXBI backplane in the slot where the T1022 module has been placed.
4. If necessary, install the transition header (PN 12-22246-01) for the slot in question using the torque screwdriver (PN 29-17381-00). Set the value on the screwdriver to 6 inch pounds. Note that the accuracy of the torque is valid only when the screw is in motion. See Figure 2-2.

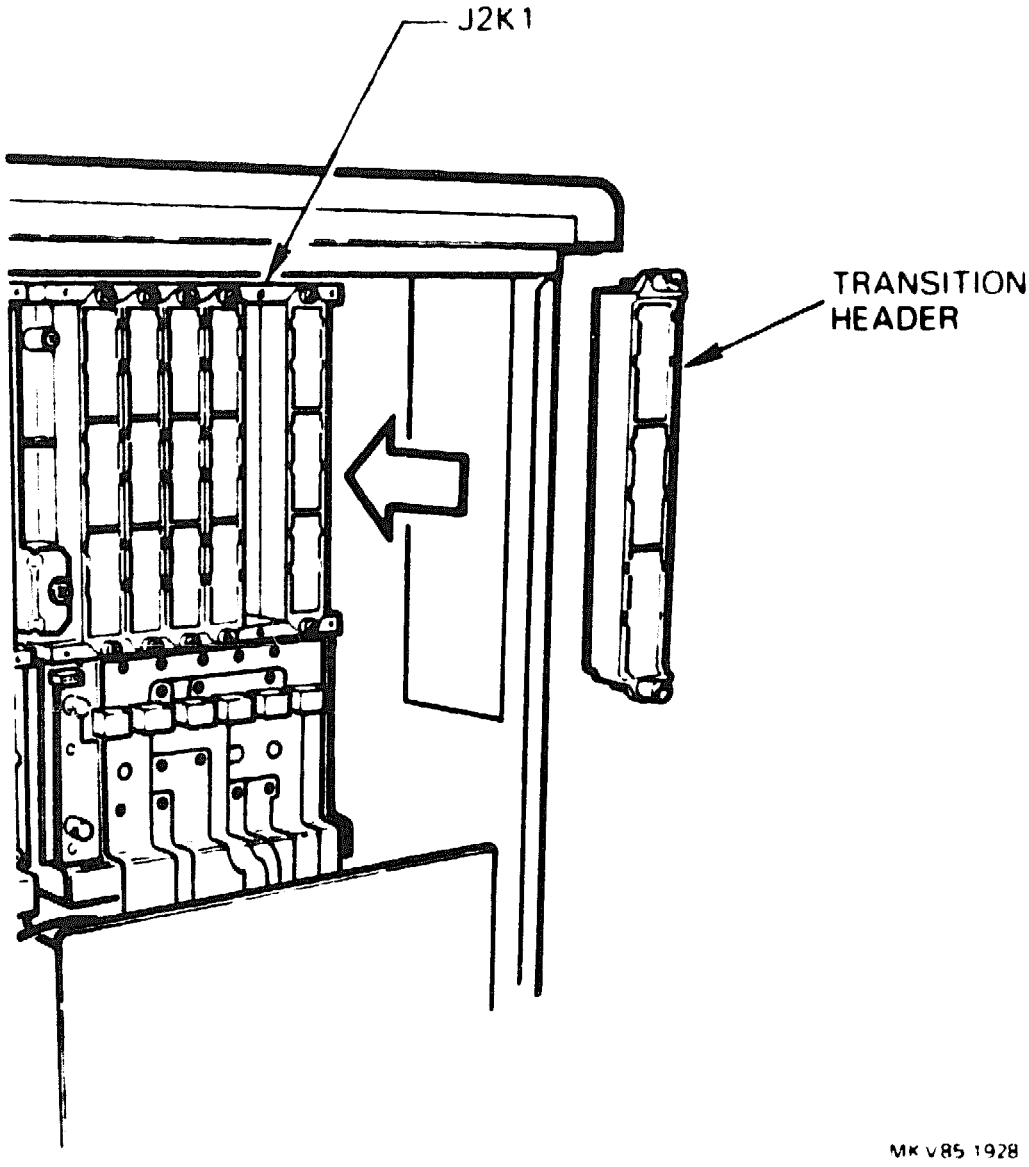
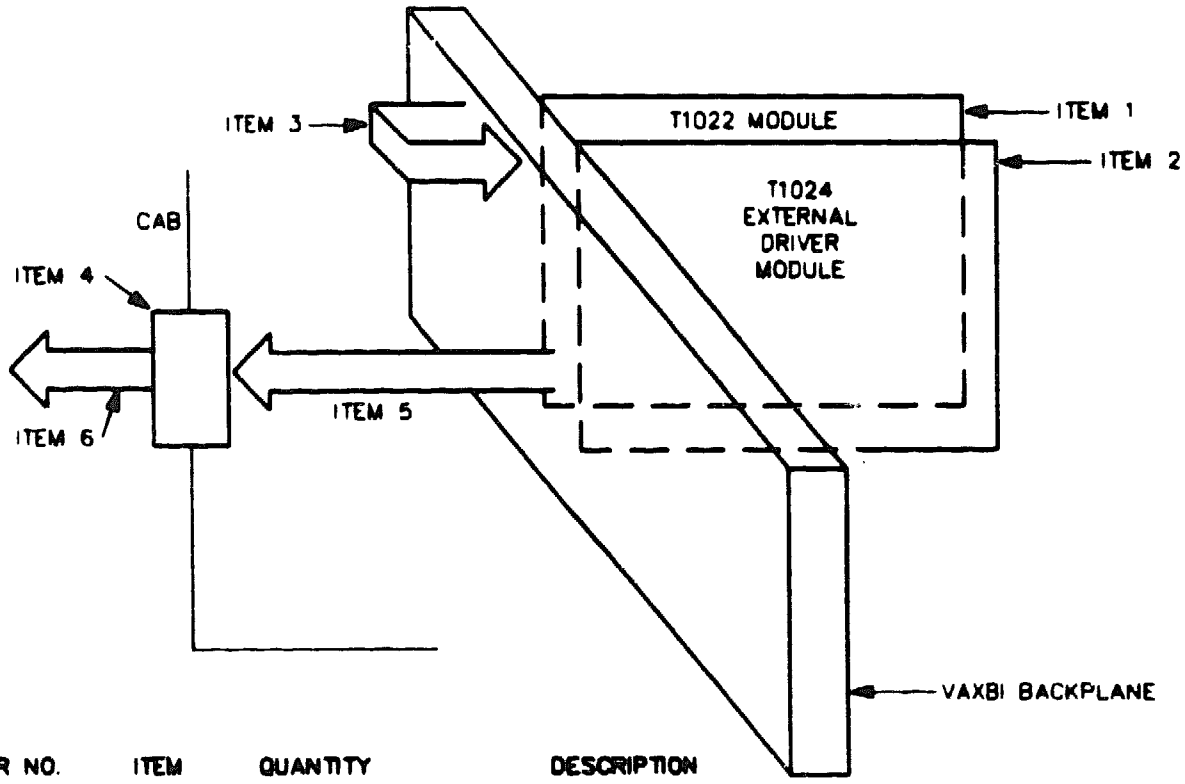


Figure 2-2 Transition Header Installation

5. Attach the three cables (CK-DRB32-L*) to section C2, D2 and E2 of the transition header. See Figure 2-4.
6. Run the cables inside the cabinet to whatever location the customer designates. Use common cable management practices as described in the System Installation Guide when performing this operation.
7. Continue the installation by following the steps in the test and verification section of this manual.

2.2.2 DRB32-E Installation

Figure 2-3 shows how the DRB32-E fits together mechanically. Use the figure and the following instructions to install the option.



ORDER NO.	ITEM	QUANTITY	DESCRIPTION
DRB32-E	1	1 EACH	DRB32-M (T1022)
	2	1 EACH	EXTERNAL DRIVER MODULE (T1024)
17-01474-01	3	3 EACH	INTERMODULE CONNECTOR CABLES
70-23924-02	4	1 EACH	I/O CONNECTOR PANEL
CK-DRB32-L*	5	1 SET (3)	INTERNAL CABLES (UP TO 15 FT)
BS17Y-20	6	1 SET (3)	EXTERNAL CABLES (20 FT)
OR			
BS17Y-40	6	1 SET (3)	EXTERNAL CABLES (40 FT)

BU-2693

Figure 2-3 DRB32-E Option

Follow the steps below for installing the DRB32-E.

1. Expose the back of the VAXBI card cage.

2. Insert the DRB32 (T1022) module into an empty VAXBI slot, following the configuration rules in section 2.1.3. All VAXBI modules are keyed to prevent incorrect installation.
3. Insert the external driver module (T1024) into the slot to the left of the T1022 module.
4. Ensure that a unique node ID plug is present on the VAXBI backplane in the slot where the T1022 module has been placed.
5. If necessary, install two transition headers (PN 12-22246-01) for the slots in question using the torque screwdriver (PN 29-17381-00). Set the value on the screwdriver to 6 inch pounds. Note that the accuracy of the torque is valid only when the screw is in motion. See Figure 2-2: Transition Header Installation.
6. Place the three intermodule cables (part number 17-01474-01) on sections C, D and E of the transition header connecting the two modules. The transition headers are split vertically and the cables are plugged into adjacent sections of both slots. Refer to Figure 2-4: Backplane and Transition Headers. Cables go from C2 of the T1022 slot to C1 of the T1024 slot, from D2 of the T1022 slot to D1 of the T1024 slot, and from E2 of the T1022 slot to E1 of the T1024 slot.
7. Install the I/O connector panel to the I/O section of the system.

WARNING
It is essential that all devices be connected to the same line-power bus to ensure that all devices have the same chassis ground. A danger to personnel and a fire hazard exists when various devices have different chassis ground potentials.

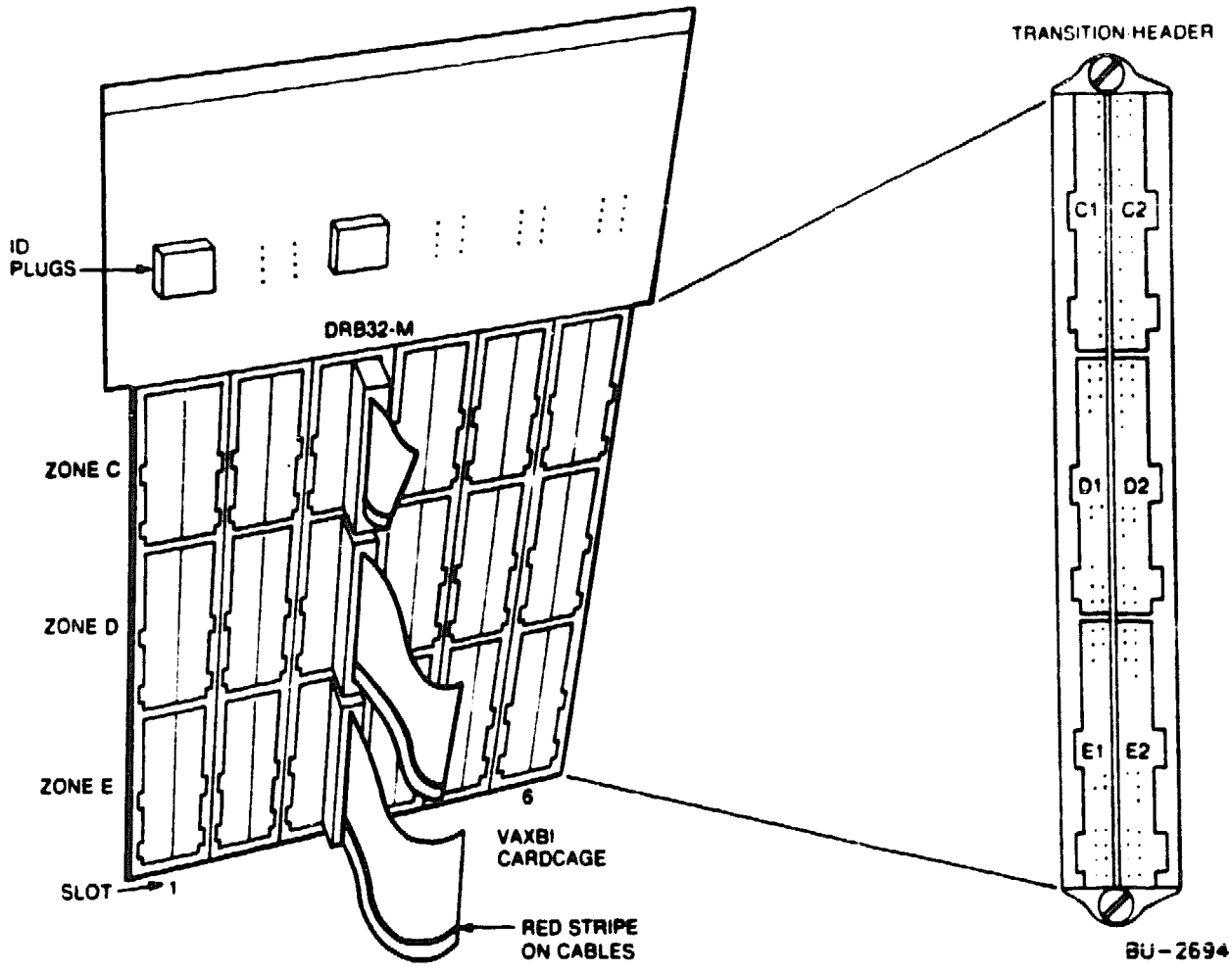


Figure 2-4 Backplane and Transition Headers

8. If only one DRB32-E is being installed in the system, follow the instructions in (a) below. If two DRB32-Es are being installed, follow the instructions in (b). Refer to Figure 2-5: DRB32-E I/O Connector Panel.

CAUTION
Cables CK-DRB32-L have a ground bus bar. Incorrect installation causes the DRB32 bus to malfunction. Ensure that the red stripes on the cables are installed on pin 30, as shown in Figure 2-4.

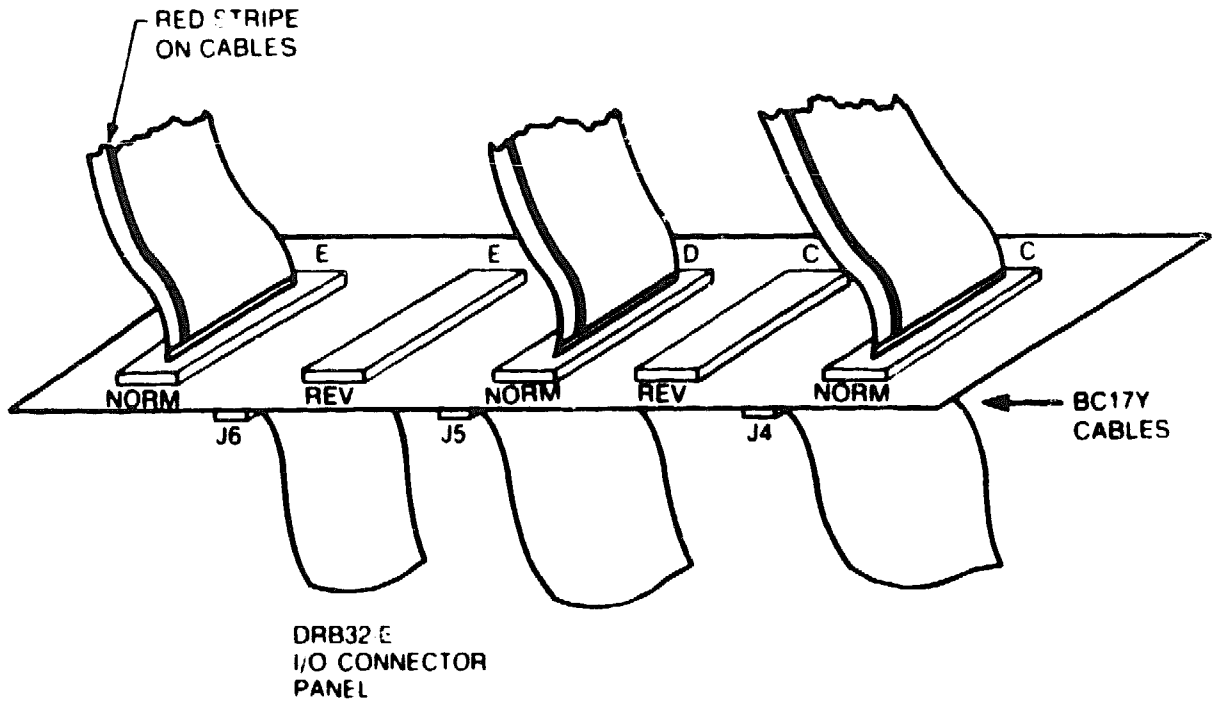


Figure 2-5 DRB32-E I/O Connector Panel

a. Install the three cables (part number CK-DRB32-L*), from the transition header of the external driver module to the I/O connector panel. Cables go from C2 of the T1024 slot to C NORM on the I/O connector panel, from D2 of the T1024 slot to D NORM on the I/O connector panel, and from E2 of the T1024 slot to E NORM on the I/O Panel. See Figure 2-4 and Figure 2-6: Typical Configuration for the DRB32-E Option.

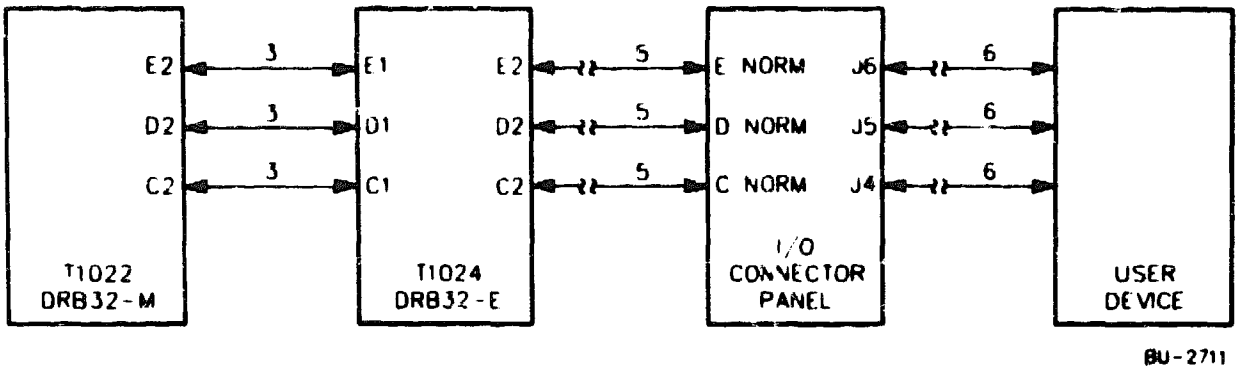


Figure 2-6 Typical Configuration for the DRB32-E Option

b. Install the three cables for one DRB32-E as in (a) above. Install the three cables for the second DRB32-E from the transition header of the second external driver module to the second I/O connector panel. Cables go from C2 of the second T1024 slot to C REV on the I/O connector panel, from D2 of the second T1024 slot to D NORM on the I/O connector panel, and from E2 of the second T1024 slot to E REV on the I/O connector panel. See Figure 2-4: Backplane and Transition Headers and Figure 2-7: Configuration for Back-To-Back DRB32-E Options.

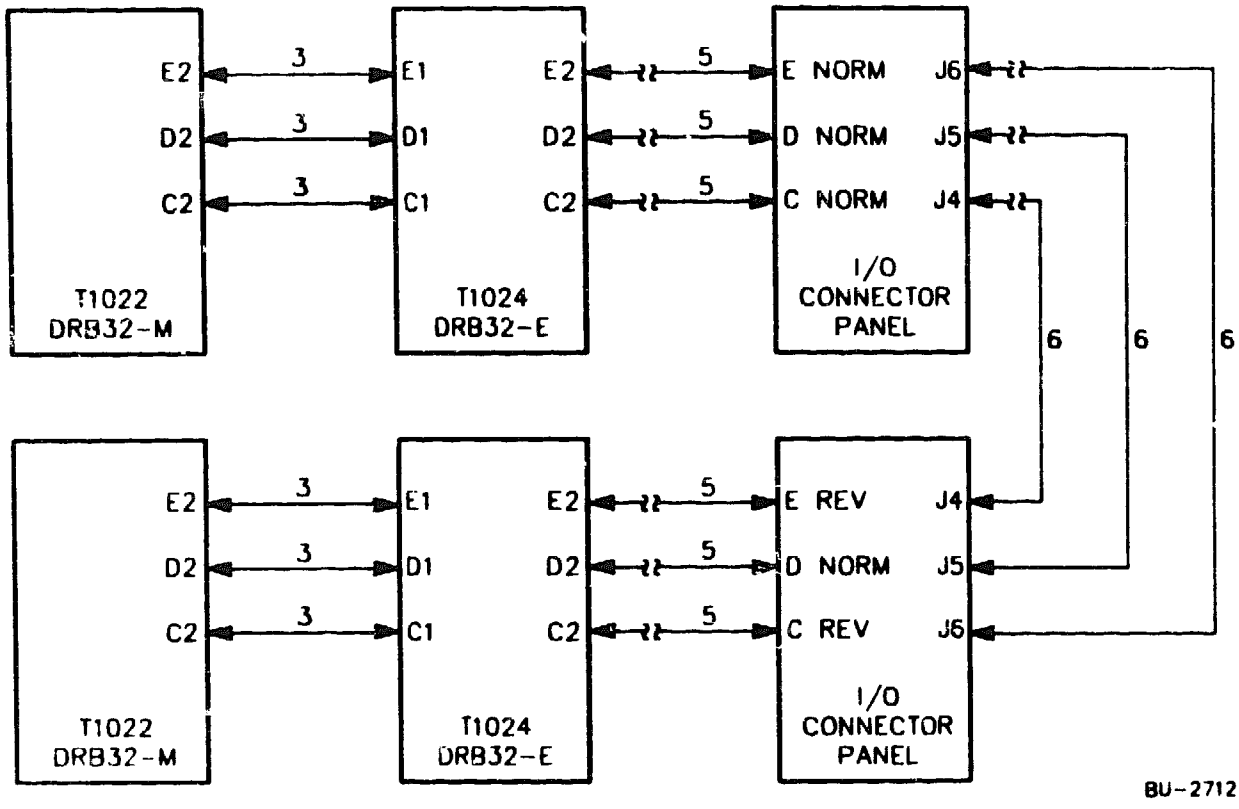


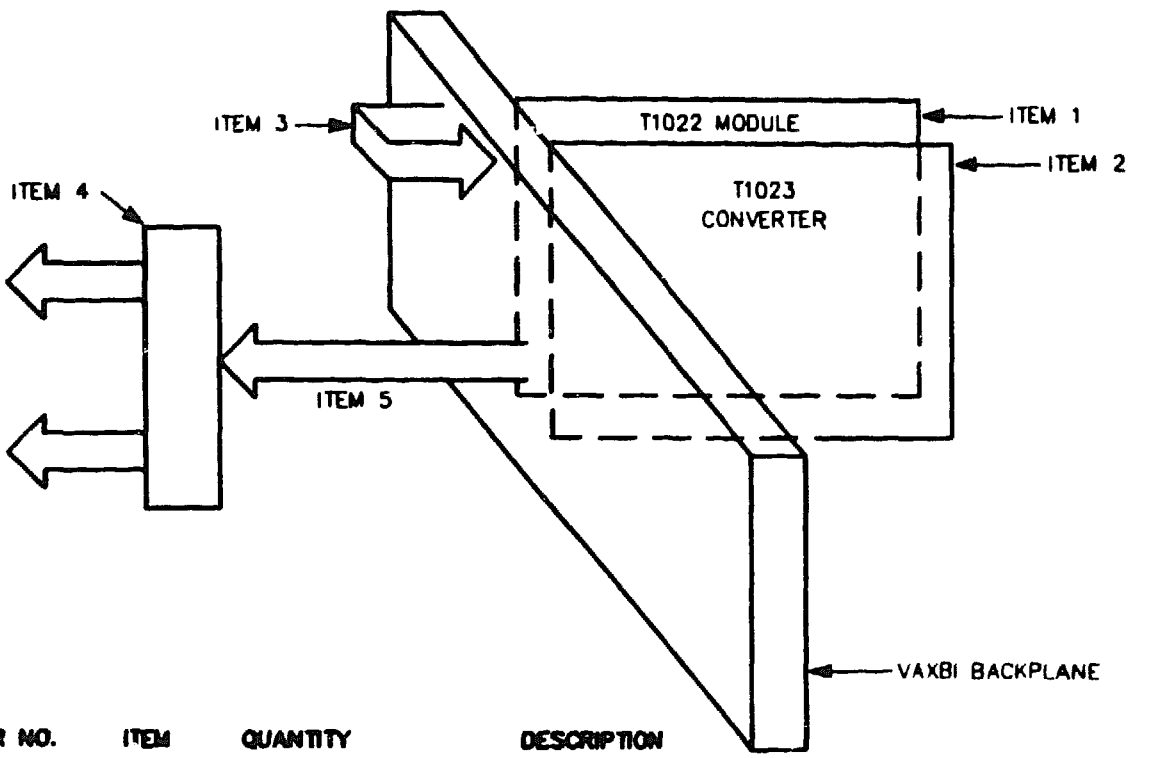
Figure 2-7 Configuration for Back-To-Back DRB32-E Options

9. Run the cables inside the cabinet between the transition header and the I/O connector panel. Use common cable management practices as described in the System Installation Guide when performing this operation.
10. Install the BS17Y-20 or -40 cables from the outside of the I/O connector panel to whatever location the customer designates using common cable management.
11. Continue the installation by following the steps in the test and verification section of this manual.

2.2.3 DRB32-W Installation

Figure 2-8 shows how the DRB32-W fits together mechanically. Use the figure and the following instructions to install the option.

NOTE
The DRB32-W went through a redesign. For an explanation of the two DRB32-W versions, please refer to the *DRB32 Technical Manual* (EK-DRB32-TM-003), Appendix C. Both the DRB32-W versions are installed, tested, and verified the same way. If you follow the steps listed in sections 2.2.3 and 3.4, you will have completed the DRB32-W option installation.



ORDER NO.	ITEM	QUANTITY	DESCRIPTION
DRB32-W	1	1 EACH	DRB32-W (T1022)
	2	1 EACH	DRB32-W CONVERTER MODULE (T1023)
17-01474-01	3	3 EACH	INTERMODULE CONNECTOR CABLES
70-23923-01	4	1 EACH	BULKHEAD CONNECTOR
CK-DRB32-L*	5	1 SET (3)	INTERNAL CABLES (UP TO 15 FT)

BU-2696

Figure 2-8 DRB32-W Option

Use the following steps to install the DRB32-W option.

1. Expose the back of the VAXBI card cage.
2. Insert the DRB32 (T1022) module into an empty VAXBI slot, following the configuration rules in section 2.1.3. All VAXBI modules are keyed to prevent incorrect installation.
3. Insert the converter module (T1023) into the slot to the left of the T1022 module.
4. Ensure that a unique node ID plug is present on the VAXBI backplane in the slot where the T1022 module has been placed.
5. If necessary, install two transition headers (PN 12-22246-01) for the slots in question using the torque screwdriver (PN 29-17381-00). Set the value on the screwdriver to 6 inch pounds. Note that the accuracy of the torque is valid only when the screw is in motion. See Figure 2-2: Transition Header Installation.
6. Place the three intermodule cables (part number 17-01474-01) on sections C, D and E of the transition header connecting the two modules. The transition headers are split vertically and the cables are plugged into adjacent sections of both slots. Refer to Figure 2-4: Backplane and Transition Headers. Cables go from C2 of the T1022 slot to C1 of the T1023 slot, from D2 of the T1022 slot to D1 of the T1023 slot, and from E2 of the T1022 slot to E1 of the T1023 slot.
7. Install the bulkhead connector to a convenient location on the cabinet frame.
8. Install the three cables (part number CK-DRB32-L*) from the transition header of the T1023 to the bulkhead connector. Cables go from C2 of the T1023 slot to C on the bulkhead connector, from D2 of the T1023 slot to D on the bulkhead connector, and from E2 of the T1023 slot to E on the bulkhead connector. See Figure 2-9: DRB32-W Bulkhead Connector.

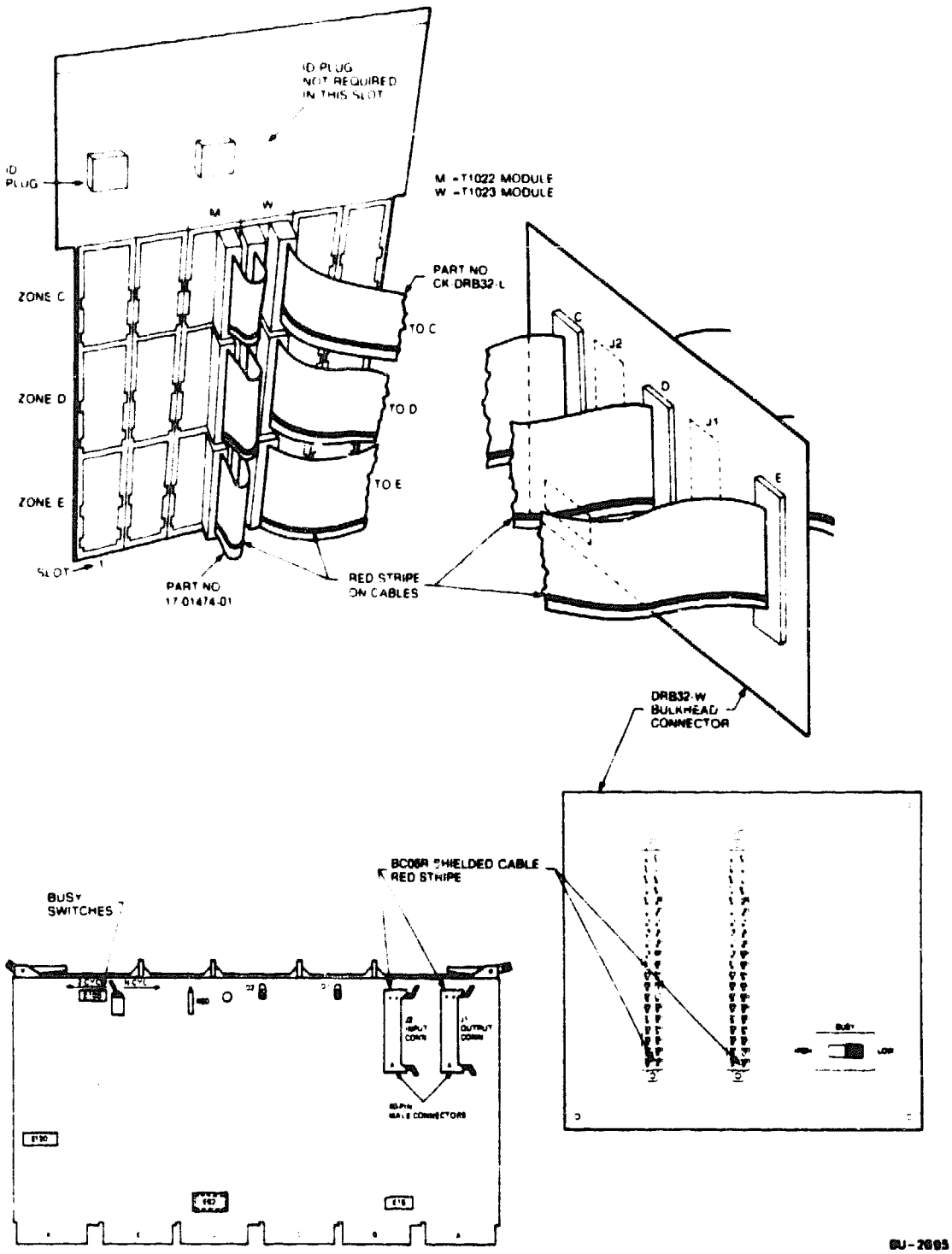


Figure 2-9 DRB32-W Bulkhead Connector

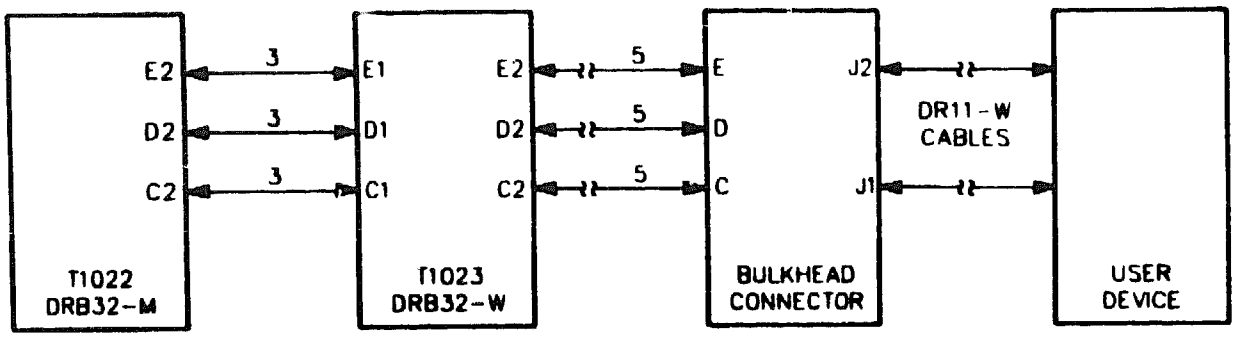
9. If the DR11-W cables are being removed from an existing DR11-W, refer to Figure 2-9 and disconnect the DR11-W cable from DMA module J1 and connect it to the bulkhead connector J1 so that what was DMA module J1-pin A goes to bulkhead connector J1-pin 1 (which forces what was DMA module J1-pin VV to go to J1-pin 40). Then, disconnect the DR11-W cable from DMA module J2 and connect it to the bulkhead connector J2 so that what was DMA module J2-pin A goes to bulkhead connector J2-pin 1 (which forces what was DMA module J2-pin VV to go to bulkhead connector J2-pin 40).

CAUTION
When using BC06R-XX shielded cables, ensure that the red stripe is connected from DR11-W pin VV to bulkhead connector pin 40 (down) so that the shield is connected to ground. See Figure 2-9. If the cables were connected "red stripe up," at the bulkhead connector, then the shield would be connected to data bit 15 (D<15>H) and signal integrity for bit 15 would be lost.

10. If the DR11-W cables are being removed from an existing DR11-W, note the position of the DR11-W switches 1 and 2. Set the BUSY switch on the bulkhead connector as follows:

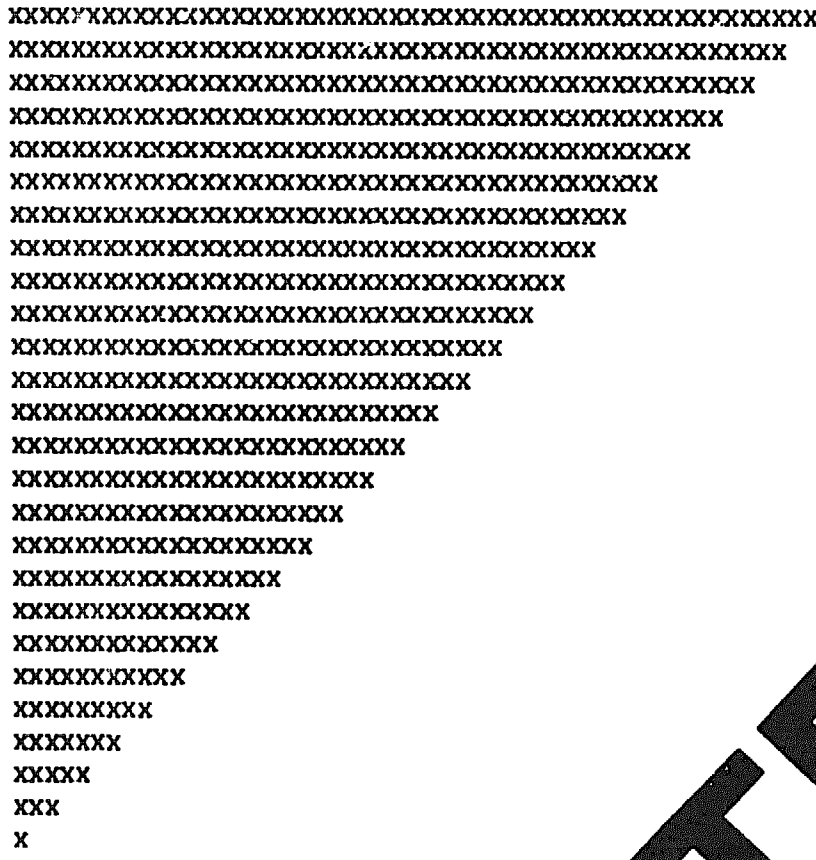
Bulkhead Connector Switch	DR11-W Switches	
BUSY	1	2
HIGH	ON	OFF
LOW	OFF	ON

11. Continue the installation by following the steps in the test and verification section of this manual.

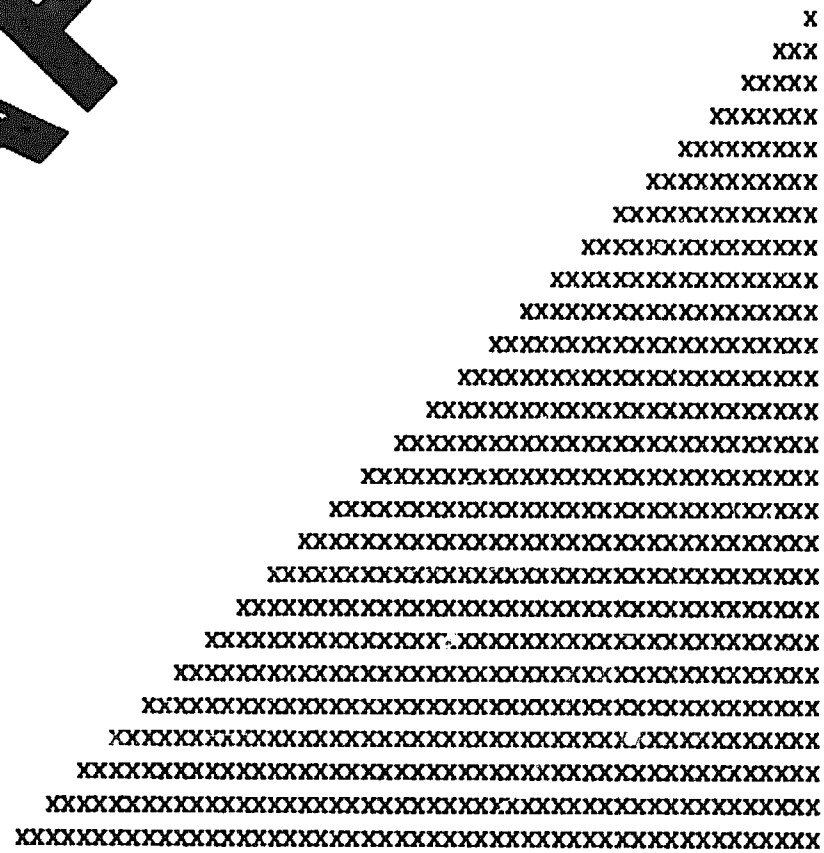


BU-2692

Figure 2-10 Typical Configuration for the DRB32-W Option



CHAPTER 3



3 Test and Verification

This chapter includes information on:

- Testing and verification of the DRB32 hardware
- The DRB32 diagnostic tools
- Troubleshooting

If the customer has ordered the DRB32 software, a VMS layered product called DRB32VMSDRIVERS, the included *DRB32 Programmer's Manual* describes the software Installation Verification Procedure (IVP) in Chapter 2.

3.1 Test and Verification

To test and verify the installation of the DRB32-M, the DRB32-E, and the DRB32-W, run the following tests.

1. Self-Test (runs on power-up)
2. EVDRH (default for the -M module)
3. EVDRH (DRB32_EXTERNAL)—External Loopback mode. Use the DRB32 output tester for the DRB32-M and DRB32-E.
4. EVDRI (default for the DRB32-W module)
5. EVDRI (DRBW_EXTERNAL)—External Loopback mode. Use the DRB32-W external loopback cable for the DRB32-W.

Self-test only exercises the DRB32-M module (the basic DRB32 board). Although the yellow LEDs light on both the DRB32-E external driver module and the DRB32-W converter module, no self-test is actually performed. On these two modules, the LED is tied to +5 volts (so that a quick inspection of the LEDs does not result in your swapping

out a good module). The LEDs on these two modules light even if the DRB32-M has not passed its internal self-test. Use EVDRH to test the DRB32-M, DRB32-E, and cables. EVDRI tests the DRB32-W.

By default, the EVDRH puts the DRB32 into Internal Loopback mode. When this occurs, the DRB32 bus drivers are disabled and will NOT put signals out onto the DRB32 bus. However, the rest of the DRB32 logic (excluding the bus drivers) is tested. EVDRH can also be run in an External Loopback mode that, with the use of the DRB32 output tester, tests the bus drivers and the cables. See the section on troubleshooting for details. Installation verification includes both default and external loopback tests. These diagnostic tests are described in the following sections.

3.2 Self-Test

Self-test for the DRB32-M (T1022) is located in ROM from address 160000 octal to 171776 octal and is executed by the T-11 processor on the DRB32. Successful completion of the DRB32-M self-test lights the yellow LED on the DRB32-M. For the VAX 8200, the console indicates which VAXBI nodes have passed self-test, by printing their node numbers. Nodes that have failed self-test are indicated by printing their node number preceded by a minus sign.

Neither the DRB32-E external driver module nor the DRB32-W converter module have self-tests on board; their LEDs are lit by tying them to +5 volts. The LEDs on the DRB32-E and DRB32-W light whether or not the DRB32-M has passed its internal self-test. (The purpose of these LEDs is to prevent mistakes in swapping out a DRB32-E or DRB32-W module that works.)

3.2.1 Normal Self-Test

Self-test uses the tests shown in Table 3-1 to exercise the logic on the DRB32-M. Note that tests 9 and A test two functions.

Table 3-1 DRB32 Self-Test Tests

Test Number	Name	Function
1	T-11 Processor	Tests function of T-11 chip by exercising instruction set, address modes, data paths, stack, registers, and other functions
2	ROM Checksum	Tests for "stuck at" faults in data and address paths of T-11 subsystem to ROM (where self-test resides)

Table 3-1 DRB32 Self-Test Tests (Continued)

Test Number	Name	Function
3	T-11 RAM	Tests T-11 RAM subsystem by alternately writing and reading As and 5s through ascending and descending addresses
4	CB RAM	Tests DRB32 Central Bus RAM by alternately writing and reading As and 5s through ascending and descending addresses
5	Map Registers	Tests DRB32 map registers by alternately writing and reading As and 5s through ascending and descending addresses
6	BCAI Register	Tests BCAI chip registers by alternately writing and reading As and 5s through ascending and descending register addresses
7	Map Port BIIC Response	Tests DRB32 map port by writing and reading data through port to CB RAM using VAXBI loopback transactions
8	Map Port VAXBI Response	Tests DRB32 map port by writing and reading data through port to CB RAM using VAXBI transaction
9	DMA Port / CSC Interrupt	Tests two functions: 1) Tests DMA port by doing DMA data transfer from CB RAM to IODAT register (using CURRENT map registers) and DMA data transfer from IODAT register to CB RAM (using the NEXT map registers) 2) Tests that T-11 can be interrupted (when interrupts are enabled) when a current segment transfer on DMA port is completed
A	Physical Address Parity/Error Interrupt	Tests two functions: 1) Tests that a (generated) parity error is detected during DMA transfer on address portion of a transfer 2) Tests that T-11 can be interrupted (when interrupts are enabled) when an error is detected on the DRB32
B	Data Parity Error	Tests that a (generated) data parity error is detected on data during loopback transactions
C	VAXBI Interrupt	Tests VAXBI interrupts

Table 3-1 DRB32 Self-Test Tests (Continued)

Test Number	Name	Function
D	Control Bus Interrupt	Tests that an interrupt is generated (when interrupts are enabled) when a change of state on the 8-bit input control lines is detected
E	Register Test	Tests DRB32 registers not used in any test above

If self-test fails, General Purpose Register 0 (at address bb+F0) contains the test number in the high order word and the error number in the low order word. For example, refer to Figure 3-1 and let us assume that a DRB32 residing at node 6 (base address 6000C000) has failed self-test on power up. By examining GPR0, we can see that test 7 has failed and with an error number of 2. The direct cause of the error can quickly be traced through the self-test listing (if self-test passes, GPR0 returns a 0).

```
#ABCDEFGHIJK..N#

. . . 3 . . -6 . 8 . . . . . .
00200000
?03
      PC + 00000002
>>>e 6000c0f0
      P       6000C0F0      0007002
>>>
```

Figure 3-1 GP0 Register

Refer to Appendix D for the Self-Test Error Code List. Self-test does VAXBI loopback transactions and reads and writes the module's own node space, but does not do the entire range of VAXBI transactions possible to the DRB32. Therefore, self-test does not test the interaction of the DRB32 with any other device on the VAXBI.

NOTE
No self-test is performed on either the DRB32-E external driver module or the DRB32-W converter module, even though an LED is lit on these boards.

3.2.2 Fast Self-Test

Fast self-test is executed when the Fast Self-Test bit in the T-11 CB RAM location register is cleared. Fast self-test only executes Test 7 and Test 9. (See Table 3-1 above.)

3.3 EVDRH (VAX Level 3 Functional Diagnostic)

EVDRH is a VAX level 3 diagnostic that runs under the control of the VAX Diagnostic Supervisor (VDS). Since EVDRH runs in the system CPU on the VAXBI, VAXBI transactions are possible, enabling further testing of the DRB32 microcode and hardware. (There is no plan for a DRB32 VAX level 2 diagnostic, since the DRB32 device driver is likely to be developed/modified by the customer.)

Since the DRB32 is used as an interface to user devices, it is not advisable for a diagnostic to put data onto the DRB32's output bus, because such data might damage the user device. Therefore, EVDRH (and also self-test) tri-states the DRB32's drivers, preventing signals from being transmitted onto the output bus. However, you can test the output buses and the DRB32 drivers with the use of the DRB32 output tester, which is described in the next two sections.

EVDRH includes the tests shown in Table 3-2.

Table 3-2 DRB32 Level 3 Tests

Test Number	Name	Function
1	DRB32 Control and Device Type Registers	Tests that device type corresponds to the DRB32
2	BIIC and DRB32 Self-Test Check	Tests that BIIC and DRB32 both passed self-test
3	BIIC and DRB32 Required Registers	Tests required registers on DRB32 are accessible, fields that can be written are writable, fields that can be read are readable
4	DRB32 CB RAM March	Tests DRB32 Central Bus RAM by alternately writing and reading As and 5s through ascending and descending addresses
5	32-Bit VAXBI-to-Port Octaword Transfer	Tests if DRB32 can read data from VAXBI memory using octaword transfers on VAXBI
6	32-Bit Port-to-VAXBI Octaword Transfer	Tests if DRB32 can write data to VAXBI memory using octaword transfers on VAXBI

Table 3-2 DRB32 Level 3 Tests (Continued)

Test Number	Name	Function
7	Page Overflow	Tests if data can be transferred to and from DRB32 across page boundaries in VAXBI memory
8	Byte Offset	Tests if data byte can be transferred to and from DRB32, regardless of its location in the two DRB32 octaword buffers
9	32-Bit VAXBI-to-Port Block Transfer	Tests that data blocks can be transferred to and from DRB32 to VAXBI memory, looping back data on DRB32
10	Parity Error	Tests if parity errors are detected on both address and data
11	Dual Transfer	Tests continuous transfer feature by using both CURRENT and NEXT registers to transfer data to and from DRB32
12	Control Bus Interrupt	Tests if VAXBI interrupt can be generated by DRB32's CTRL IN lines
13	Error Interrupt	Tests if VAXBI interrupt request is generated by DMA transfer error
14	Single-System LINK Cable Test	Tests ability of two DRB32s in same system to communicate via their cable link
15	Single-System LINK Data Test	Tests ability of two DRB32s in same system to transfer data to each other
16	DRB32 External Loopback Test	Tests integrity of the DRB32 data path (drivers, receivers, cables); test requires use of DRB32 Output Tester and its loopback cable
17	Dual-System (cab) Link	Tests ability of two DRB32-Es in different systems to communicate; can be used to verify that DRB32s work together

The EVDRH diagnostic has several sections and section names. These section names (and the tests that each section performs) are shown in Table 3-3.

Table 3-3 EVDRH Section Names

Section Names	Tests Performed
1) DEFAULT	(All DRB32 internal loopback tests) Tests 1 - 13
2) SELFTEST_CONTROL	Allows control of self-test: can cause self-test to loop on error, loop on test, halt on error, etc.
3) DRB32_EXTERNAL	Runs DRB32 external loopback test
4) SINGLE_CAB_LINK	Runs LINK test with two DRB32s in a single-system
5) MASTER_DUAL_CAB_LINK	Runs LINK test with two DRB32s in different systems
6) SLAVE_DUAL_CAB_LINK	Runs LINK test with two DRB32s in different systems

3.3.1 Running EVDRH

To run EVDRH, first boot the VAX Diagnostic Supervisor (VDS), (for specific VDS boot procedures, refer to the appropriate processor technical manual), load the diagnostic, attach the device(s) under test, and start the appropriate section.

Figures 3-2 through 3-4 are examples of the terminal output produced when you attach a DRB32 for the internal loopback test section of EVDRH, run the test, and have an error.

```
                    Copyright Digital Equipment Corporation
                          1986, 1987, 1988, 1989, 1990
                               All Rights Reserved.

DIAGNOSTIC SUPERVISOR,  ZZ-EBSAA-X12.6-1996   8-MAR-1990 03:43:20
DS> load evdrh
****************************************************************************

                    Copyright Digital Equipment Corporation
                          1986, 1987, 1988, 1989,1990
                               All Rights Reserved.

                        CONFIDENTIAL DIAGNOSTIC SOFTWARE
                                PROPERTY OF
                         DIGITAL EQUIPMENT CORPORATION

              Use authorized only pursuant to a valid Right-to Use License.

****************************************************************************
DS> att
Device type? drb32
Device Link? hub
Device Name? uq1
Node? 6
BR? 4
DS> set tra,h
DS> sel all
DS>
```

Figure 3-2 Attaching a DRB32 for Internal Loopback Testing

Once the DRB32 has been attached and selected, the internal loopback (default) tests can be run as shown in Figure 3-3.

```
DS> st

.. Program: EVDRH - DRB32-M, -E Functional Diagnostic, revision 1.1,
16 tests, at 00:00:57.09.
Testing: _UU1

Test 1: DRB32 Control And Device Type Registers Test
Test 2: BIIC and DRB32 Self Test Check Test
Test 3: BI/DRB32 Required Registers Test
Test 4: DRB32 CB RAM March Test
Test 5: 32 Bit BI to Port Octaword Transfer Test
Test 6: 32 Bit Port to BI Octaword Transfer Test
Test 7: Page Overflow Test
Test 8: Byte Offset Test
Test 9: 32 Bit BI to Port Block Transfer Test
Test 10: Parity Error Test
Test 11: Dual Transfer Test
Test 12: Control Bus Interrupt Test
.. End of run, 0 errors detected, pass count is 1,
   time is 8-MAR-1990 03:48:48.88
DS>
```

Figure 3-3 Selecting Default Tests

If the DRB32 is malfunctioning, an error is output to the user during the failing test. This is illustrated in Figure 3-4.

```
DS> st/te:5

.. Program: EVDRH - DRB32-M, -E Functional Diagnostic, revision 1.1,
16 tests, at 00:10:01.49.
Testing: _UQ1

Test 5: 32 Bit BI to Port Octaword Transfer Test

******** EVDRH - DRB32-M, -E Functional Diagnostic - 1.1 ********
 Pass 1, test 5, subtest 0, error 6,  1-JAN-1985 00:10:14.67
 Hard error while testing UQ1: DRB32 ERROR

BI TO IODAT REGISTER TRANSFER ERROR.

EXPECTED VALUE OF IODAT: 55555555(X)

RECEIVED VALUE OF IODAT: AAAAAAAA(X)


******** End of Hard error number 6 ********


.. Halt on error at PC 00007A50(x)
DS>
```

Figure 3-4 Test Failure Example

3.3.2 DRB32 Output Tester

NOTE
The DRB32 Output Tester can only be run by a qualified Digital representative.

The DRB32 output tester is a receiving device to test the DRB32's output drivers and cables without damaging user devices or causing unpredictable events. Since the DRB32 is a half-duplex device, the DRB32 output tester provides a means of testing hardware without sending data out to user devices. The DRB32 output tester is used with EVDRH. To test the DRB32's output bus, you must disconnect the user device. The DRB32 output tester can be attached to either an I/O connector panel, the VAXBI backplane, or to cables running from the backplane to the I/O connector panel, thus providing a way to isolate malfunctioning cables or drivers.

The DRB32 output tester consists of 37 LEDs mounted on a PC card in a box, with connector plugs on the top. Provided cables are plugged into these connectors and allow testing of the:

1. Output drivers on the DRB32-M
2. Cable between the DRB32-M and the DRB32-E or a user device
3. Output drivers on the DRB32-E
4. Cable between the DRB32-E and the I/O connector panel
5. Signals coming from the DRB32-E's I/O connector panel
6. Output of the FCC compliant cables that run from the I/O connector panel to the user device.

A drawing of the DRB32 output tester LED placement is shown in Figure 3-5.

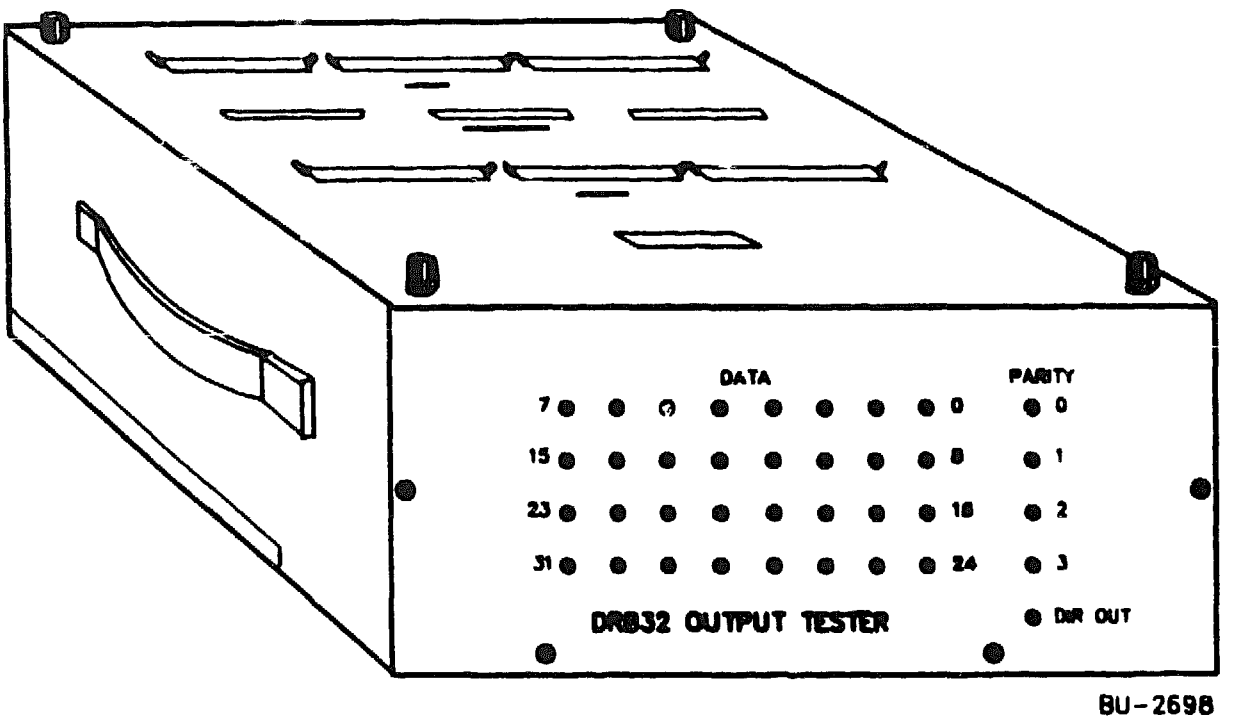


Figure 3-5 DRB32 Output Tester

Thirty-two LEDs are tied to bits 0–31 on the output bus. Four LEDs are tied to the parity bits associated with each byte of data, and one LED is tied to the signal DIR OUT. Each LED is labeled on the Tester. The Control signals on the control bus are looped back (CTRL OUT L <0-7> to CTRL IN L <0-7>). CTRL OUT <7> is also tied to DIR IN so that direction on the bus may be established.

The external loopback test diagnoses, first, the data path to the DRB32 transceivers and then the data path from the transceivers to the outside world. The DRB32 output tester is used only when the data path to the transceivers is known to be good. If it is not, an error message is reported and the DRB32 output tester portion of the test does not run. To run the external loopback test, simply attach the DRB32 (if you have not already done so previously) and type:

```
DS> STA/SEC=DRB32_EXTERNAL
```

Figure 3-6 shows the instructions that appear on your terminal for the data line test if the data path to the transceivers is good. After you type "GO" to proceed with the test and before you press RETURN to display the first pattern, ensure that the tester is displaying "OK", as shown in the top of Figure A-6.

```
DS> sta

.. Program: EVDRH - DRB32-M, -E Functional Diagnostic, revision 1.1,
16 tests, at 00:32:34:02.

Testing: _UQ1    _UQ2

Test 16: DRB32 External Loopback Test Routine

WARNING: Make sure the loopback connector, cables, etc. are connected
         properly.  Failure to do so could result in damage to the user
         equipment.

Type ''GO'' when you are ready to proceed with the test [GO] go

                         PART I: Data Lines Test

        This test attempts to verify the integrity of the 32
        data lines which interface the DRB32 directly to the
        outside world. Pay close attention to only those LEDs
        which comprise the data field; the parity LEDs (right-
        most) can be ignored for now. The object of the test
        is to ensure that each adjacent LED (e.g., data lines)
        will respond completely independent of any other LED.
        If not, then a short or open exists at the data line in
        question.

        NOTE: The patterns will be output to the loopback connector
        in 6 sets of 2 patterns each. This is to facilitate easy
        viewing and quick verification of the integrity of the data
        lines.

Press RETURN when ready to proceed with the test [<CR>]:

                <Pattern set number 1 is now being displayed>

Press RETURN when ready to proceed with the test [<CR>]:

                <Pattern set number 2 is now being displayed>

Press RETURN when ready to proceed with the test [<CR>]:

                <Pattern set number 3 is now being displayed>

Press RETURN when ready to proceed with the test [<CR>]:

                <Pattern set number 4 is now being displayed>

Press RETURN when ready to proceed with the test [<CR>]:

                <Pattern set number 5 is now being displayed>

Press RETURN when ready to proceed with the test [<CR>]:

                <Pattern set number 6 is now being displayed>

Press RETURN when ready to proceed with the test [<CR>]:
```

Figure 3-6 Data Lines Test

The instructions in Figure 3-7 appear on your terminal for the parity lines test.

```
                    PART II: Parity Lines Test

        This test attempts to verify the integrity of the 4 parity
        lines (rightmost) associated with the 32 data lines. As in
        the previous test, the object is to ensure that each parity
        line will respond independent of any other parity line.
        This is accomplished by running two 1s through a field of
        0s. The important thing is to verify that the parity lines
        respond by reading, alternately, 1010 and 0101


Press RETURN when ready to proceed with the test [<CR>]:


                    <The parity lines test patern is now being
                     output to the loopback connector>

Press RETURN to advance to the next pattern or to exit the test:


                    <The parity lines test pattern is not being
                     output to the loopback connector>

Press RETURN to advance to the next pattern or to exit the test:
```

Figure 3-7 Parity Lines Test

Appendix A shows the LED data patterns. Patterns alternate, so that you can tell if a signal is bad by looking at the patterns. If an error occurs prior to the LED panel test, a message similar to the one in Figure 3-8 is output to the user and the test ends.

```
DS> sta

.. Program: EVDRH - DRB32-M, -E Functional Diagnostic, revision 1.1,
16 tests, at 00:30:33.28.

Test 16: DRB32 External Loopback Test Routine

WARNING:
        Make sure the loopback connector, cables, etc, are connected
        properly, Failure to do so could result in damage to user
        equipment.

Type ''GO'' when you are ready to proceed with the test [GO] go

******** EVDRH - DRB32-M, -E Functional Diagnostic - 1.1 ********
 Pass 1, test 1, subtest 0, error 3,  8-MAR-1990 03:48:48:81
 Hard error while testing UQ1: DRB32 ERROR

BI TO IODAT REGISTER TRANSFER ERROR.

EXPECTED VALUE OF IODAT: FFFFFFFF(X)

RECEIVED VALUE OF IODAT: 00EFEF6F(X)

******** End of Hard error number 3 ********
```

Figure 3-8 Error on Testing

3.4 EVDRI (VAX Level 3)

EVDRI is also a VAX level 3 diagnostic that runs under the control of the VAX Diagnostic Supervisor (VDS). This diagnostic tests the functionality and data path of the DRB32-W converter module. The DRB32-W is a device that interfaces the DRB32 (a VAXBI device) to the DR11-W (a UNIBUS device).

Like EVDRH, EVDRI comprises two distinct program sections: an internal loopback section and an external loopback section. The external loopback test requires the use of an external loopback cable (part no. 70-23923), but unlike the connector for EVDRH there are no LEDs associated with it. The two program sections contain identical tests, but the external loopback test provides maximum hardware coverage. EVDRI includes the following tests.

Table 3-4 EVDRI Tests

Test Number	Name	Function
1	DRB32-W Internal Loopback Longword DMA Test	Tests the ability of the DRB32-W to transfer longwords via DMA transactions
2	DRB32-W Internal Loopback Byte DMA Test	Tests the ability of the DRB32-W to transfer bytes via DMA transactions
3	DRB32-W Internal Loopback ATTN H Interrupt Test	Tests the ability of the DRB32-W to respond to and clear an ATTN H interrupt
4	DRB32-W Internal Loopback Direction Error Interrupt Test	Tests the ability of the DRB32-W to respond to a direction error interrupt and clear the error condition
5	DRB32-W Internal Loopback DATAIP Error Interrupt Test	Tests the ability of the DRB32-W to respond to a Data In Pause error interrupt and clear the error condition
6	DRB32-W External Loopback Longword DMA Test	Tests the ability of the DRB32-W to transfer longwords via DMA transactions
7	DRB32-W External Loopback Byte DMA Test	Tests the ability of the DRB32-W to transfer bytes via DMA transactions
8	DRB32-W External Loopback EVEN Word Transfer Test	Tests the reliability of the DRB32-W word counter functionality during EVEN word transfers
9	DRB32-W External Loopback ODD Word Transfer Test	Tests reliability of the DRB32-W word counter functionality during ODD word transfers
10	DRB32-W External Loopback ATTN H Interrupt Test	Tests the ability of the DRB32-W to respond to and clear ATTN H interrupt
11	DRB32-W External Loopback Direction Error Interrupt Test	Tests the ability of the DRB32-W to respond to a direction error interrupt and clear the error condition
12	DRB32-W External Loopback DATAIP Error Interrupt Test	Tests the ability of the DRB32-W to respond to a Data In Pause error interrupt and clear the error condition

Table 3-4 EVDRI Tests (Continued)

Test Number	Name	Function
13	DRB32-W Single-System LINK Test T1023-YA	Tests the ability of two DRB32-Ws (T1023-YA) to pass messages in PIO and DMA
14	DRB32-W Single-System LINK Test T1023-00	Tests the ability of two DRB32-Ws (T1023-00) to pass messages in PIO and DMA

EVDRI has two sections and section names. These section names (and the tests that each section performs) are shown in Table 3-5.

Table 3-5 EVDRI Sections

Section Names	Tests Performed
1) DEFAULT	(All DRB32-W internal loopback tests) Tests 1-5
2) DRBW_EXTERNAL	Runs DRB32-W external loopback tests Tests 6-12
3) DRBW_SINGLE_SYSTEM_LINK	Runs DRB32-W single-system LINK test Tests 13 and 14

NOTE
T1023-YA is a new version of the DRB32-W. The T1023-00 is the old (T1023-AA) version. Only the new version is shipped, but these diagnostics will test any old version (T1023-00) module previously installed in the system.

3.4.1 Running EVDRI

Before diagnosing the DRB32-W using EVDRI, be sure the DRB32-M module is fully functional. Only after the DRB32-M is known to be good should diagnosing of the DRB32-W take place. This will help prevent erroneous testing results. To run the default section (internal loopback tests) of EVDRI, first make sure the DRB32-W and the DRB32-M are attached by the appropriate cables. Boot the VAX Diagnostic Supervisor, load EVDRI, and follow the attach sequence (section 3.3.1) for the DRB32-M module. The DRB32-W does not contain a VAXBI corner and, therefore, does not need to be attached to the supervisor process. Type STA and the DRB32-W internal loopback tests are then run and appear as shown in Figure 3-9.

```
                Copyright Digital Equipment Corporation
                     1986, 1987, 1988, 1989, 1990.
                         All Rights Reserved.

DIAGNOSTIC SUPERVISOR,  ZZ-EBSAA-X12.6-1996   8-MAR-1990 03:43:20
DS> load evdri
********************************************************************************

                Copyright Digital Equipment Corporation
                     1986, 1987, 1988, 1989, 1990.
                         All Rights Reserved.

                    CONFIDENTIAL DIAGNOSTIC SOFTWARE
                              PROPERTY OF
                     DIGITAL EQUIPMENT CORPORATION

        Use authorized only pursuant to a valid Right-to-Use License.

********************************************************************************
DS> att
Device type? drb32
Device Link? hub
Device Name? uq1
Node? 6
BR? 4
DS> set tra,h
DS> sel all
DS> sta

.. Program: EVDRI - DRB32-W Functional Diagnostic, revision 3.0, 14
   tests, at 03:44:13.99.
Testing: UQ1

Test 1: DRB-W Internal Loopback Longword DMA Test
Test 2: DRB-W Internal Loopback Byte DMA Test
Test 3: DRB-W Internal Loopback ATTNH Interrupt Test
Test 4: DRB-W Internal Loopback Direction Error Interrupt Test
Test 5: DRB-W DATAIP Error Test
.. End of run, 0 errors detected, pass count is 1,
   time is  8-MAR-1990 03:48:48.40
```

Figure 3-9 DRB32-W Internal Loopback Test Example

If an error occurs during testing, an error message (similar to the one in Figure 3-10) is output to the user.

```
******** EVDRI - DRB32-W, Functional Diagnostic - 1.1 ********
 Pass 1, test 1, subtest 0, error 10,  8-MAR-1990 03:48:48.40
 Hard error while testing UQ1: DRB32 ERROR

A DMA READ DOES NOT MATCH THE DATA WRITTEN TO THE DRB-W
EXPECTED: 00000000
RECEIVED: 00400040

******** End of Hard error number 10 ********
```

Figure 3-10 DRB32-W Test Error

3.4.2 DRB32-W Loopback Cable

For complete coverage of the DRB32-W, it is necessary to disconnect the DRB32-W from the user equipment and connect it to a loopback cable (part no. BC051-1C). To use the DRB32-W loopback cable, you must set a switch on the bulkhead connector to the low position to change the polarity of the BUSY H signal.

NOTE
If the switch is not set properly, the diagnostic reminds the user; see Figure 3-11.

```
The BUSY bit on the DRBW connector is not set. Please set the
switch on the external loopback connector before continuing.
```

Figure 3-11 Improperly Set Switch Message

An illustration of the DRB32-W loopback cable is shown in Figure 3-12.

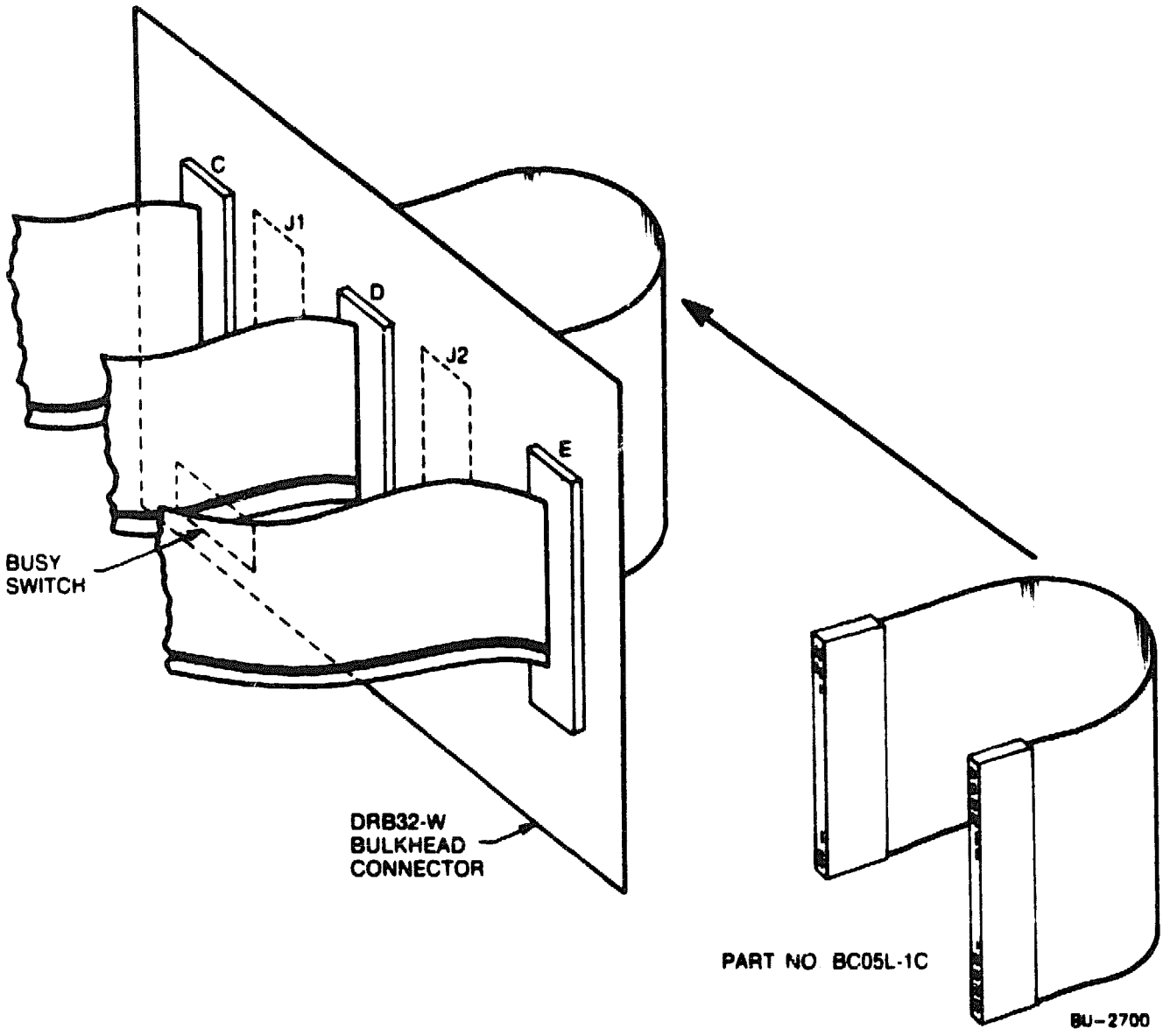


Figure 3-12 DRB32-W Loopback Cable

The external loopback test is run by calling out the specific section in the diagnostic. This is illustrated in Figure 3-13.

```
DS> sta/sec=drbw_external

.. Program: EVDRI- DRB32-W Functional Diagnostic, revision 3.0, 14
   tests, at 03:44:13.99.
Testing: _UQ1

Test 6: DRB-W External Loopback Longword DMA Test
Test 7: DRB-W External Loopback Byte DMA Test
Test 8: DRB32-W External Loopback EVEN Word Transfer Test
Test 9: DRB32-W External Loopback ODD Word Transfer Test
Test 10: DRB32-W External Loopback ATTN H Interrupt Test
Test 11: DRB32-W External Loopback Direction Error Interrupt Test
Test 12: DRB32-W External Loopback DATAIP Error Interrupt Test
.. End of run, 0 errors detected, pass count is 1,
   time is  8-MAR-1990 03:48:51.40
```

Figure 3-13 External Loopback Test Example

If an error occurs, a message to that effect is output to the user. For an example of a sample error message you might expect to see, refer to the error message example under running the DRB32-W internal loopback tests, Figure 3-10.

3.5 Troubleshooting

Follow general systemwide techniques for troubleshooting with this option. Some suggestions are:

- Self-test and module swap works in most cases
- Seating in the VAXBI backplane is important
- Torque settings on transition headers are critical; use 6 inch pounds—greater torque causes damage to the transition header and backplane
- Always use a grounded wrist strap to prevent static damage to the modules

3.5.1 Diagnosing the DRB32-M

The DRB32 has two Field Replaceable Units:

1. The T1022 module
2. The cables between the VAXBI backplane and the user device.

Self-test and EVDRH default sections test the DRB32-M module logic up to the bus drivers. If both these test pass, but the customer is not able to use the option, either the drivers or the cable is faulty. Use the DRB32 output tester to isolate the failure to either the cables or the DRB32 drivers.

If customer failures persist, and efforts to correct hardware problems do not work, there may be a problem with customer-produced hardware or software.

3.5.2 Diagnosing the DRB32-E

The DRB32-E consists of:

1. The DRB32-M (T1022) module
2. The DRB32-E (T1024) module
3. Sets of cables

   Between the two modules on the VAXBI backplane

   From the VAXBI backplane to the I/O connector panel

   From the I/O connector panel to the user device
4. The I/O connector panel

There is no self-test on the T1024 module. Its LED is tied to +5 volts so that when this LED is lit, it does not indicate that self-test passed. The module is designed to compensate for differential grounds between systems/devices not in the same electrical environment as the DRB32. It is primarily made up of ECL logic and has a very high MTBF. To troubleshoot this option, you must use EVDRH with the DRB32 output tester to isolate the failing part.

3.5.3 Diagnosing the DRB32-W

The DRB32-W consists of:

1. The DRB32-M (T1022) module
2. The DRB32-W (T1023) module
3. Sets of cables

   Between the two modules on the VAXBI backplane

   From the VAXBI backplane to the bulkhead connector

   From the bulkhead connector to the user device
4. The bulkhead connector

There is no self-test on the T1023 module. Its LED is tied to +5 volts so that when this LED is lit, it does not indicate that it passed self-test. This module is designed to provide interface capabilities from the DRB32 to a DR11-W device.

To troubleshoot this option, you must use EVDRI in Internal Loopback mode and External Loopback mode with the loopback cable. Ensure that the DRB32-M module has been thoroughly verified to be working so as to eliminate this from the diagnostic path. The quickest method of diagnosing the DRB32-W converter module is to run the Internal Loopback mode tests. If any of these tests fails, the module or the cables connecting the DRB32-M to the DRB32-W converter module are faulty. Replace the cables and run the tests again. If the tests fail this time, the DRB32-W converter module is at fault.

If these tests pass, you should run the external loopback tests. If any one of these tests fails, the DRB32-W module or the cables to the bulkhead connector are bad. To isolate the problem, use known good cables from the DRB32-W converter module to the bulkhead connector. If the tests now fail, the DRB32-W is faulty. Conversely, if the tests pass, replace the cables.

3.6 Message Passing Between DRB32s (LINK Tests)

The purpose of the DRB32 link tests is to test the cable interconnect between two VAXBI systems containing DRB32 hardware. The diagnostic coverage is minimal, though the link test does exploit a great deal of the transfer capabilities of the DRB32. For this reason, the LINK test should not be run if the DRB32 internal/external tests have not previously run successfully. Only after the other tests have verified the functionality of the connected DRB32s are the results of the LINK test meaningful.

There are two versions of the LINK test:

- Single-system LINK test
- Dual-system LINK test

The single-system LINK test is used to test two DRB32s that are in the same system. This test is especially useful for manufacturing testing. The dual-system LINK test is used to test two DRB32s devices in different systems.

3.6.1 Running the EVDRH Single-System LINK Test

Use the EVDRH single-system LINK test (or intrasystem LINK test) to functionally test two DRB32s in the same VAXBI system. The LINK test is only meaningful after self-test and all level 3 internal/external loopback tests have been run successfully.

The EVDRH single-system LINK test actually consists of two subtests:

- The first test verifies that the cables are connected properly by writing and reading the DRB32's IOCTL and IODAT registers.
- The second test exchanges protocol and data between the two DRB32s in a manner similar to an actual application. The total data transfer is variable, up to approximately 1MB of data.

To run the EVDRH single-system LINK test, follow these instructions:

- Place two DRB32s as close to each other as possible in the same VAXBI backplane (be careful to note at which nodes they reside). Using any available internal cables (see Table 2-2: Cable Ordering Numbers) or the intermodule cable (17-01474-01) from the output tester, connect the DRB32s together as explained in section 2.2.2 for the T1022 and T1024 modules.
- Boot the supervisor and ATTACH the DRB32s.
- Set trace and select the units to test.
- Set event flag 1 to attach two UUTs for simultaneous testing.
- Run the first test.

See Figure 3-14 for an example of running the single-system LINK test (first test).

```
                    Copyright Digital Equipment Corporation
                         1986, 1987, 1988, 1989, 1990.
                              All Rights Reserved.

DIAGNOSTIC SUPERVISOR,  ZZ-EB5AA-X12.6-1996   8-MAR-1990 03:43:20
DS> l evdrh
****************************************************************

                    Copyright Digital Equipment Corporation
                         1986, 1987, 1988, 1989, 1990.
                              All Rights Reserved.

                       CONFIDENTIAL DIAGNOSTIC SOFTWARE
                                PROPERTY OF
                        DIGITAL EQUIPMENT CORPORATION

              Use authorized only pursuant to a valid Right-to Use License.

****************************************************************
DS> att
Device type? drb32
Device Link? hub
Device Name? uq1
Node? 6
BR? 4
DS> att
Device type? drb32
Device Link? hub
Device Name? uq2
Node? 7
BR? 4
DS> set tra
DS> sel all
DS> set ev fl1
```

Figure 3-14 Running EVDRH Single-System LINK Test

If the cables are connected properly and the DRB32 drivers are fully functional, the output appears as in Figure 3-15.

```
Test 1: Link Modules IOCTL/IODAT Test
... End of run, 0 errors detected, pass count is 1.
```

Figure 3-15 Running Single-System LINK Test: Test 1

If an error occurs, a typical run might look like the one shown in Figure 3-16.

```
DS>

.. Program: EVDRH - DRB32-M, -E Functional Diagnostic, revision 1.1,
16 tests, at 03:48:50.40.
Testing: _UQ1    _UQ2

Test 14: Link Modules IOCTL/IODAT Test
********  EVDRH - DRB32-M, -E Functional Diagnostic - 1.1 ********
 Pass 1, test 14, subtest 0, error 1,  0-MAR-1990 03:48:48.40
 Hard error while testing UQ1: DRB32 ERROR

ERROR TRANSFERRING DATA FROM ''ORIGINAL MASTER'' IOCTL TO ''ORIGINAL
SLAVE'' IOCTL.

EXPECTED:
 00, 01, 02, 03, 04, 05, 06, 07, 08, 09, 0A, 0B, 0C, 0D, 0E, 0F

 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 1A, 1B, 1C, 1D, 1E, 1F

 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 2A, 2B, 2C, 2D, 2E, 2F

 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 3A, 3B, 3C, 3D, 3E, 3F

 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 4A, 4B, 4C, 4D, 4E, 4F

 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 5A, 5B, 5C, 5D, 5E, 5F

 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 6A, 6B, 6C, 6D, 6E, 6F

 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 7A, 7B, 7C, 7D, 7E, 7F

 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 8A, 8B, 8C, 8D, 8E, 8F

 90, 91, 92, 93, 94, 95, 96, 97, 98, 99, 9A, 9B, 9C, 9D, 9E, 9F

 A0, A1, A2, A3, A4, A5, A6, A7, A8, A9, AA, AB, AC, AD, AE, AF

 B0, B1, B2, B3, B4, B5, B6, B7, B8, B9, BA, BB, BC, BD, BE, BF

 C0, C1, C2, C3, C4, C5, C6, C7, C8, C9, CA, CB, CC, CD, CE, CF

 D0, D1, D2, D3, D4, D5, D6, D7, D8, D9, DA, DB, DC, DD, DE, DF

 E0, E1, E2, E3, E4, E5, E6, E7, E8, E9, EA, EB, EC, ED, EE, EF

 F0, F1, F2, F3, F4, F5, F6, F7, D8, D9, FA, FB, FC, FD, FE, FF
```

Figure 3-16 Single-System LINK Test: Test 1 Error Example (Part 1 of 2)

```
RECEIVED:
FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF
FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF
FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF
FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF
FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF
FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF
FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF
FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF
FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF
FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF
FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF
FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF
FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF
FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF
FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF
FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF
*********  End of Hard error number 1 *********
```

Figure 3-16 Single-System LINK Test: Test 1 Error Example (Part 2 of 2)

If an error occurs, check to make sure the cables are connected properly. If the connection is good, the cables may be faulty. After Test 1 successfully runs to completion, run Test 2. (See Figure 3-17.)

```
DS> sta/sec-single_cab_link

.. Program: EVDRH - DRB32-M, -E Functional Diagnostic, revision 1.1,
16 tests, at 05:48:48.50
Testing: _UQ1   _UQ2

Test 14: Link Modules IOCTL/IODAT Test
Test 15: DRB32 single-system Link Test

BEGINNING INITIAL DATA TRANSFER (1000 PAGES), [ORIGINAL MASTER (UQ10)
-> ORIGINAL SLAVE (UQ2)].

THE LINK HAS BEEN TURNED AROUND. WE WILL BEGIN TRANSFERRING DATA IN
THE OPPOSITE DIRECTION.
[ORIGINAL SLAVE (UQ2) -> ORIGINAL MASTER (UQ1)]



PROGRESS_FLAG = 000A
.. End of run, 0 errors detected, pass count is 1,
   time is  8-MAR-1990 05:48:48.40
DS>
```

Figure 3-17 Single-System LINK Test: Test 2 Example

Note that the link is established and executed in two different directions. If an error occurs, the run may look like the one shown in Figure 3-18.

```
******** EVDRH - DRB32-M, -E Functional Diagnostic - 1.1 ********
 Pass 1, test 15, subtest 0, error 6,  8-MAR-1990 00:19:55.61
 Hard error while testing UQ1: DRB32 ERROR

11 DATA ERROR(S) OCCURRED TRANSFERRING LONGWORD DATA FROM THE
ORIGINAL MASTER TO THE ORIGINAL SLAVE.


******** End of Hard error number 6 ********


.. Halt on error at PC 0000AD2E(X)
DS>co
.. Continuing from 0000AD2E


DATA ERROR IN RECEIVE BUFFER OCCURRED AT ADDRESS: 000F4E00
DATA AT THIS ADDRESS : FFFFFFFF (TRANSMIT DATA: 5555AAAA
RESUMED NORMALCY AT ADDRESS: 000F4E2C
DATA AT THIS ADDRESS: 20524556  (TRANSMIT DATA): 20524556
NUMBER OF BYTES TRANSFERRED INCORRECTLY: 44


THE LINK HAS BEEN TURNED AROUND. WE WILL BEGIN TRANSFERRING DATA
IN THE OPPOSITE DIRECTION.
[ORIGINAL SLAVE (UQ2) -> ORIGINAL MASTER (UQ1)].



PROGRESS_FLAG = 000A
.. End of run, 1 error detected, pass count is 1,
   time is  8-MAR-1990 00:20:12.58
```

Figure 3-18 Single-System LINK Test: Test 2 Error Example

The error message tries to show where a data error started, and where it no longer appeared, along with the data that was transmitted.

3.6.2 Running the Dual-System LINK Test

To run the dual-system LINK test (see Figure 3-19), you need the following:

- Two VAXBI systems, each outfitted with a DRB32
- A cable set to connect the systems together through the DRB32s

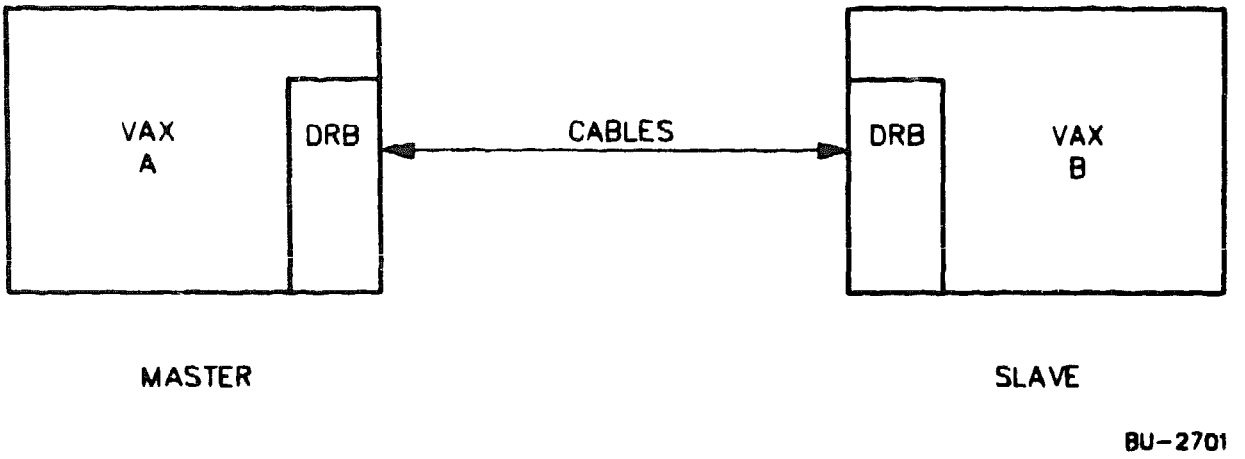


Figure 3-19 Dual-System LINK Test Configuration

Use the following steps to run the LINK test:

1. Boot the VAX Diagnostic Supervisor and load a copy of the DRB32 level 3 diagnostic (EVDRH) into each system.
2. ATTACH the DRB32s.
3. Designate one system as the MASTER and enter:

```
DS> STA/SEC = MASTER_DUAL_SYS_LINK
```

4. The other system is the SLAVE. On this system enter:

```
DS> STA/SEC = SLAVE_DUAL_SYS_LINK
```

5. Press <Return> on the MASTER's terminal to start the MASTER first, and then start the SLAVE as soon as possible.
6. The LINK test now sets up and transfers data between the systems. All status and errors are reported by means of the Diagnostic Supervisor.
7. At the completion of the test, a message similar to the following appears:

```
...End of run, 0 errors detected, pass count is 1.
time is 8-MAR-1990  11:47:18.67
```

3.6.3 Interpreting the Dual-System LINK Test Results

When the slave starts, it initializes its side of the link and waits for the master to initiate a transfer. If the master has not been started within one minute, a reminder is displayed on the slave's console, prompting the user to start the master. The prompt is repeated at one minute intervals until the master is started or a CTRL C is typed to abort the session.

When the master is started, it initializes its side to write a packet of data to the slave. This packet is a random calculation of data and transfer size. The master then attempts to write the link interface. Once the slave receives all the data, the link turns around and the slave writes the data back to the master. When the master receives the data, it compares the received packet against its transmitted buffer and reports any data compare errors. If no errors are found, the master sets up another transfer and repeats the test, for a total of 100 passes.

Figure 3-20 shows a sample run of the master console for a successful test, while Figure 3-21 shows the same from the slave console.

```
MASTER: PASS COUNT = 1, TRANSFER SIZE = 1136 bytes
MASTER: PASS COUNT = 2, TRANSFER SIZE = 4464 bytes
MASTER: PASS COUNT = 3, TRANSFER SIZE = 6896 bytes
MASTER: PASS COUNT = 4, TRANSFER SIZE = 3760 bytes
MASTER: PASS COUNT = 5, TRANSFER SIZE = 6224 bytes
MASTER: PASS COUNT = 6, TRANSFER SIZE = 3272 bytes
MASTER: PASS COUNT = 7, TRANSFER SIZE = 8136 bytes
MASTER: PASS COUNT = 8, TRANSFER SIZE = 5704 bytes
MASTER: PASS COUNT = 9, TRANSFER SIZE = 648 bytes
MASTER: PASS COUNT = 10, TRANSFER SIZE = 2280 bytes
MASTER: PASS COUNT = 11, TRANSFER SIZE = 3184 bytes
MASTER: PASS COUNT = 12, TRANSFER SIZE = 5488 bytes
MASTER: PASS COUNT = 13, TRANSFER SIZE = 7408 bytes
MASTER: PASS COUNT = 14, TRANSFER SIZE = 7600 bytes
MASTER: PASS COUNT = 15, TRANSFER SIZE = 208 bytes
MASTER: PASS COUNT = 16, TRANSFER SIZE = 4360 bytes
MASTER: PASS COUNT = 17, TRANSFER SIZE = 4520 bytes
MASTER: PASS COUNT = 18, TRANSFER SIZE = 7840 bytes
MASTER: PASS COUNT = 19, TRANSFER SIZE = 4320 bytes
MASTER: PASS COUNT = 20, TRANSFER SIZE = 1088 bytes
MASTER: PASS COUNT = 20, TRANSFER SIZE = 72 bytes
MASTER: PASS COUNT = 20, TRANSFER SIZE = 5640 bytes
MASTER: PASS COUNT = 20, TRANSFER SIZE = 5272 bytes
MASTER: PASS COUNT = 20, TRANSFER SIZE = 3928 bytes
```

Figure 3-20 Sample of a Successful Test from the Master Console

```
SLAVE: PASS COUNT = 1, TRANSFER SIZE = 1136 bytes
SLAVE: PASS COUNT = 2, TRANSFER SIZE = 4464 bytes
SLAVE: PASS COUNT = 3, TRANSFER SIZE = 6896 bytes
SLAVE: PASS COUNT = 4, TRANSFER SIZE = 3760 bytes
SLAVE: PASS COUNT = 5, TRANSFER SIZE = 6224 bytes
SLAVE: PASS COUNT = 6, TRANSFER SIZE = 3272 bytes
SLAVE: PASS COUNT = 7, TRANSFER SIZE = 8136 bytes
SLAVE: PASS COUNT = 8, TRANSFER SIZE = 5704 bytes
SLAVE: PASS COUNT = 9, TRANSFER SIZE = 648 bytes
SLAVE: PASS COUNT = 10, TRANSFER SIZE = 2280 bytes
SLAVE: PASS COUNT = 11, TRANSFER SIZE = 3184 bytes
SLAVE: PASS COUNT = 12, TRANSFER SIZE = 5488 bytes
SLAVE: PASS COUNT = 13, TRANSFER SIZE = 7408 bytes
SLAVE: PASS COUNT = 14, TRANSFER SIZE = 7600 bytes
SLAVE: PASS COUNT = 15, TRANSFER SIZE = 208 bytes
SLAVE: PASS COUNT = 16, TRANSFER SIZE = 4360 bytes
SLAVE: PASS COUNT = 17, TRANSFER SIZE = 4520 bytes
SLAVE: PASS COUNT = 18, TRANSFER SIZE = 7840 bytes
SLAVE: PASS COUNT = 19, TRANSFER SIZE = 4320 bytes
SLAVE: PASS COUNT = 20, TRANSFER SIZE = 1088 bytes
SLAVE: PASS COUNT = 20, TRANSFER SIZE = 72 bytes
SLAVE: PASS COUNT = 20, TRANSFER SIZE = 5640 bytes
SLAVE: PASS COUNT = 20, TRANSFER SIZE = 5272 bytes
SLAVE: PASS COUNT = 20, TRANSFER SIZE = 3928 bytes
```

Figure 3-21 Sample of a Successful Test from the Slave Console

Both the master and slave check for transfer errors, but only the master performs data compares. When reading data, either side reports parity errors and errors which cause timeouts to occur, such as missing sync signals. Figure 3-22 illustrates an error message on the master's console and Figure 3-23 shows an error message on the slave's console.

```
TEST 17: DRB32-M, DRB-E Dual System Link Test
******** EVDRH - DRB32-M,-E Functional Diagnostic -1.0 ********
 Pass 1, test 17, subtest 0, error 4, 8-MAR-1990 15:24:30.78
 Hard error while testing UQ1: DRB32 ERROR

 DATA COMPARE ERROR STARTING AT ADDRESS AB00.

 EXPECTED: 435FD671
 RECEIVED: 535E4771

******** End of Hard error number 4 ********

.. Halt on error at PC 00008C44(X)

DS>
```

Figure 3-22 Master Console Error Message

```
TEST 17: DRB32-M, DRB-E Dual System Link Test
******** EVDRH - DRB32-M,-E Functional Diagnostic -1.0 ********
 Pass 1, test 17, subtest 0, error 4, 8-MAR-1990 15:24:30.78
 Hard error while testing UQ1: DRB32 ERROR

 PARITY ERROR DETECTED ON READ

 ERREG = 00000009

******** End of Hard error number 2 ********

.. Halt on error at PC 0000AC50(X)

DS>
```

Figure 3-23 Slave Console Error Message

NOTE
If an error is discovered on one side of the link, the other side will usually report a timeout.

3.6.4 The DRB32-W Single-System LINK Test

The EVDRI Single-System LINK Test exercises the cable interconnection and data path between two DRB32-Ws. If it is being used in troubleshooting a faulty system, this test must be run after the entire suite of EVDRH and EVDRI internal and external tests have been successfully run. It can also be used to quickly verify a complete system after maintenance or initial installation. Use the following procedure to run the EVDRI Single-System LINK Test.

1. Boot the Diagnostic Supervisor.
2. Load the diagnostic and attach the units to be tested.
3. Set trace and select the units to be tested.
4. Set event flag 1 to attach two UUTs for simultaneous testing.
5. Start the test.

Refer to Figure 3-24 for an example of invoking the DRB32-W Single-System LINK Test and a sample printout of a successful run for the first 27 of 100 passes. Errors are reported similar to the way they are reported for the EVDRH LINK tests.

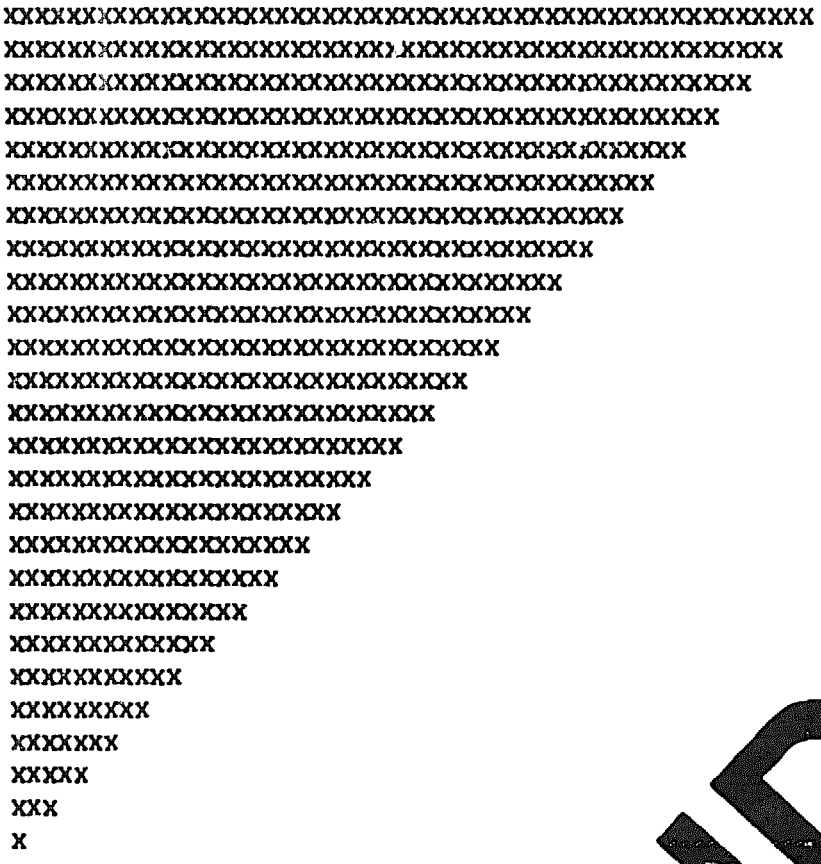
```
                Copyright Digital Equipment Corporation
                     1986, 1987, 1988, 1989, 1990.
                          All Rights Reserved.

DIAGNOSTIC SUPERVISOR,  ZZ-EBSAA-X12.6-1996   8-MAR-1990 03:43:20
DS> load evdri
*****************************************************************************

                Copyright Digital Equipment Corporation
                     1986, 1987, 1988, 1989, 1990.
                          All Rights Reserved.

                    CONFIDENTIAL DIAGNOSTIC SOFTWARE
                              PROPERTY OF
                    DIGITAL EQUIPMENT CORPORATION

     Use authorized only pursuant to a valid Right-to-Use license.

*****************************************************************************
DS> att
Device type? drb32
Device Link? hub
Device Name? uq1
Node? 6
BR? 4
DS> att
Device type? drb32
Device Link? hub
Device Name? uq2
Node? 7
BR? 4
DS> set tra
DS> sel all
DS> set ev fl 1
DS> sta/sec=drbw_single_system_link
```

Figure 3-24 Running EVDRI Single-System LINK Test (Part 1 of 2)

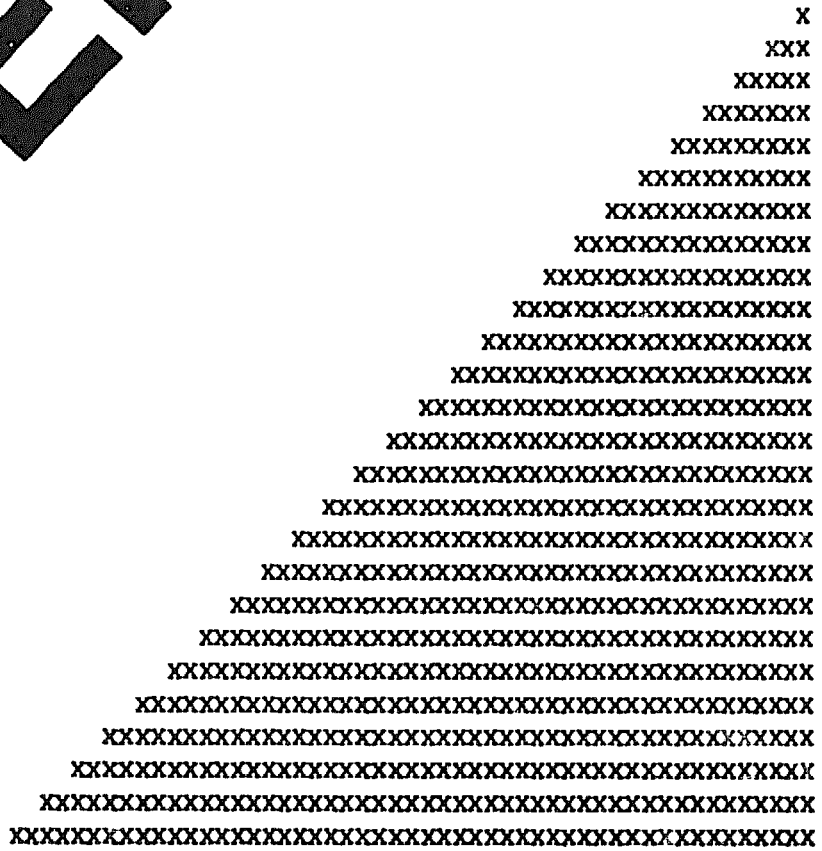
```
.. Program: EVDRI - DRB32-W Functional Diagnostic, revision 3.0, 14
   tests, at 03:44:13.99
Testing: UQ1   UQ2

Test 13: DRB32-W Single System LINK Test T1023-YA

a) Testing PIO Mode

b) Testing DMA Mode

MASTER/SLAVE: PASS COUNT = 1, TRANSFER SIZE = 1136 bytes
MASTER/SLAVE: PASS COUNT = 2, TRANSFER SIZE = 8000 bytes
MASTER/SLAVE: PASS COUNT = 3, TRANSFER SIZE = 15 bytes
MASTER/SLAVE: PASS COUNT = 4, TRANSFER SIZE = 5640 bytes
MASTER/SLAVE: PASS COUNT = 5, TRANSFER SIZE = 3272 bytes
MASTER/SLAVE: PASS COUNT = 6, TRANSFER SIZE = 4320 bytes
MASTER/SLAVE: PASS COUNT = 7, TRANSFER SIZE = 208 bytes
MASTER/SLAVE: PASS COUNT = 8, TRANSFER SIZE = 6896 bytes
MASTER/SLAVE: PASS COUNT = 9, TRANSFER SIZE = 2280 bytes
MASTER/SLAVE: PASS COUNT = 10, TRANSFER SIZE = 3928 bytes
MASTER/SLAVE: PASS COUNT = 11, TRANSFER SIZE = 1088 bytes
MASTER/SLAVE: PASS COUNT = 12, TRANSFER SIZE = 4380 bytes
MASTER/SLAVE: PASS COUNT = 13, TRANSFER SIZE = 45 bytes
MASTER/SLAVE: PASS COUNT = 14, TRANSFER SIZE = 1290 bytes
MASTER/SLAVE: PASS COUNT = 15, TRANSFER SIZE = 4367 bytes
MASTER/SLAVE: PASS COUNT = 16, TRANSFER SIZE = 5704 bytes
MASTER/SLAVE: PASS COUNT = 17, TRANSFER SIZE = 1500 bytes
MASTER/SLAVE: PASS COUNT = 18, TRANSFER SIZE = 16 bytes
MASTER/SLAVE: PASS COUNT = 19, TRANSFER SIZE = 450 bytes
MASTER/SLAVE: PASS COUNT = 20, TRANSFER SIZE = 1230 bytes
MASTER/SLAVE: PASS COUNT = 21, TRANSFER SIZE = 780 bytes
MASTER/SLAVE: PASS COUNT = 22, TRANSFER SIZE = 2500 bytes
MASTER/SLAVE: PASS COUNT = 23, TRANSFER SIZE = 72 bytes
MASTER/SLAVE: PASS COUNT = 24, TRANSFER SIZE = 7843 bytes
MASTER/SLAVE: PASS COUNT = 25, TRANSFER SIZE = 546 bytes
MASTER/SLAVE: PASS COUNT = 26, TRANSFER SIZE = 1200 bytes
MASTER/SLAVE: PASS COUNT = 27, TRANSFER SIZE = 8 bytes
```

Figure 3-24 Running EVDRI Single-System LINK Test (Part 2 of 2)

NOTE
Test 13 would be skipped if an old version of the DRB32-W (T1023-00) was installed in the system. In the case of the T1023-00 module, test 14 would be run (instead of Test 13). No user intervention is needed. The diagnostic will determine which DRB32-W (T1023-00 or T1023-YA) is in the system and will run the correct test automatically. Both DRB32-W options must be the same version in order for the link test to work correctly. If you had one old version (T1023-00) and a new version (T1023-YA) in link mode, then link mode test would fail.



APPENDIX A



A DRB32 Output Tester Patterns

The patterns below are generated in pairs by the diagnostic EVDRH. These patterns alternate at a default rate of two seconds, a rate that can be changed by the operator. To change to the next pattern, press <CR> at the diagnostic prompt.

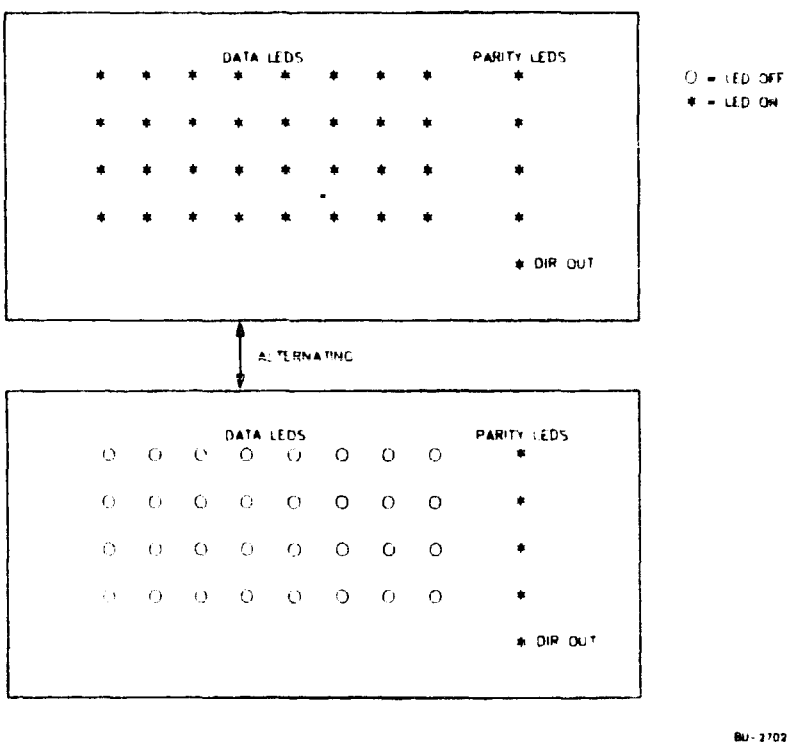


Figure A-1 Pattern 1

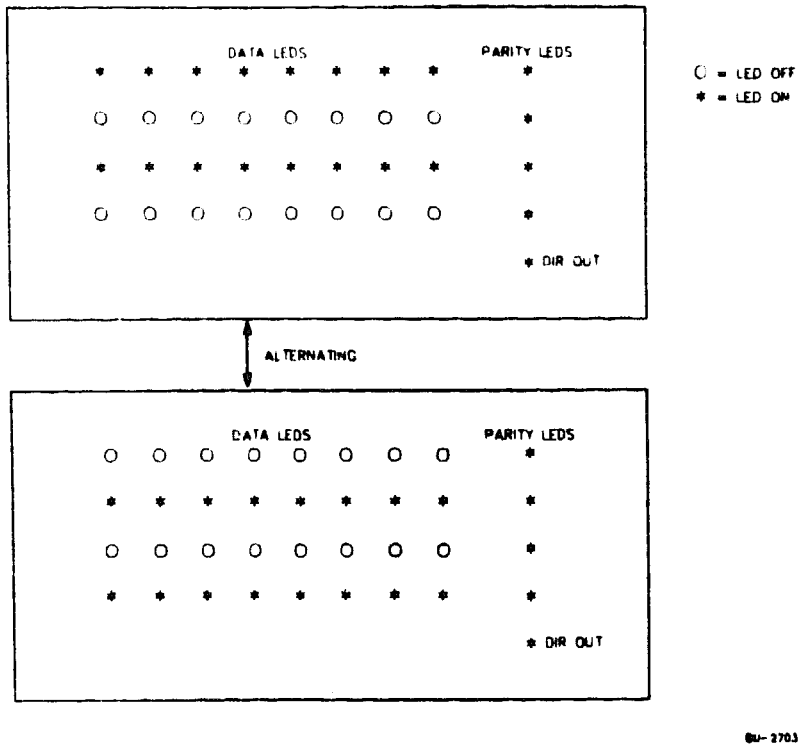


Figure A-2· Pattern 2

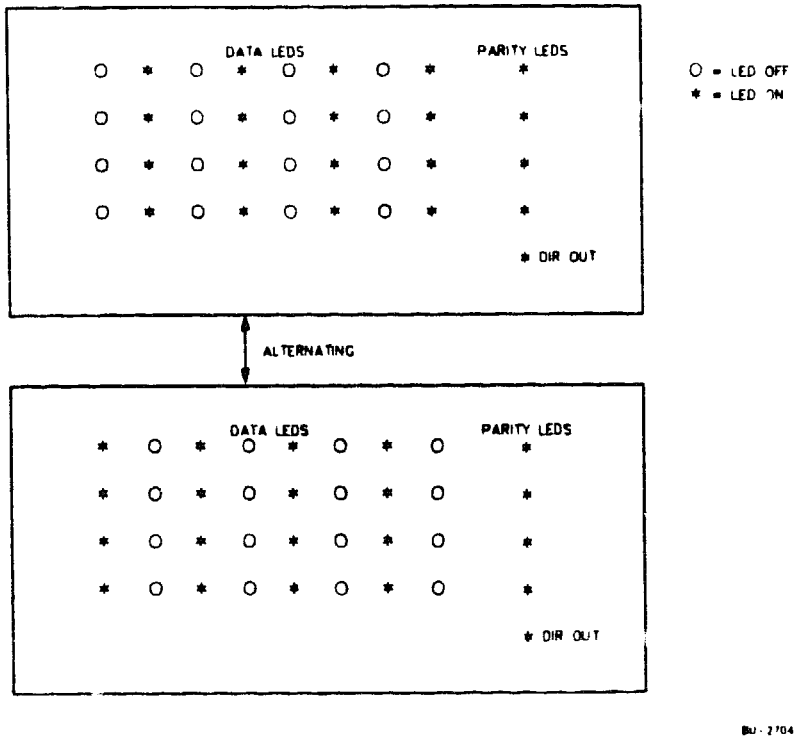
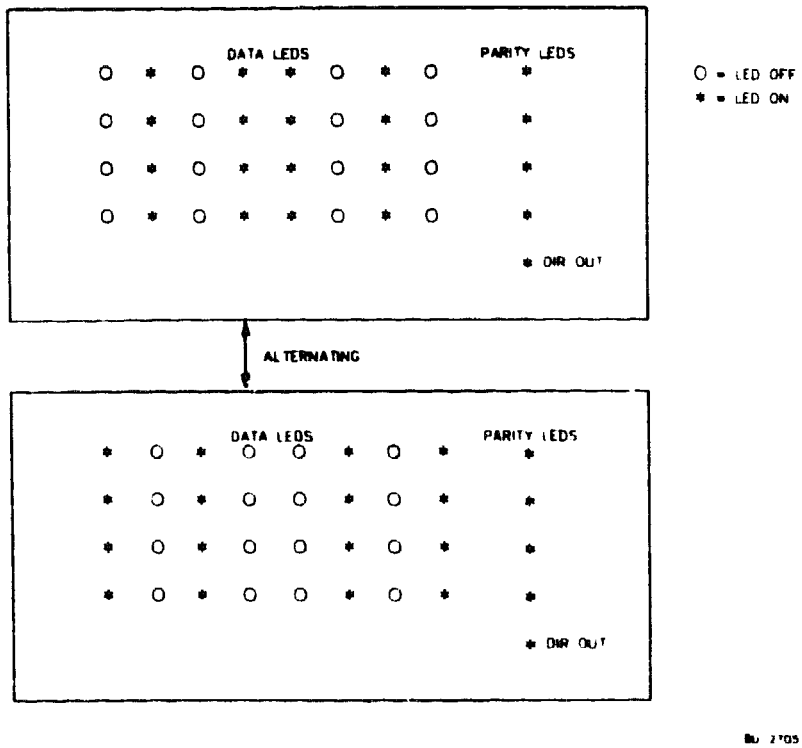
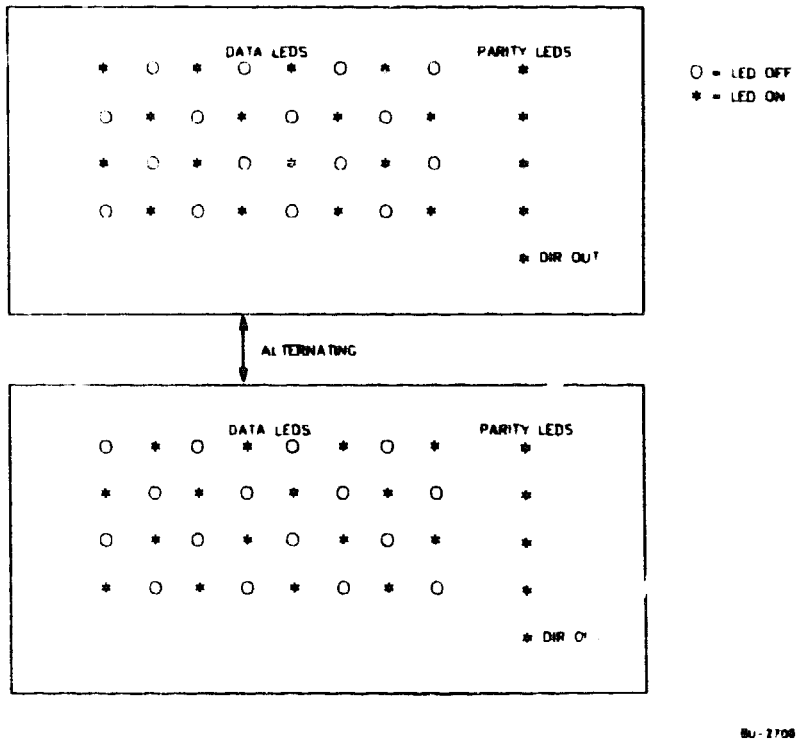


Figure A-3 Pattern 3



BU 2705

Figure A-4 Pattern 4



BU-2708

Figure A-5 Pattern 5

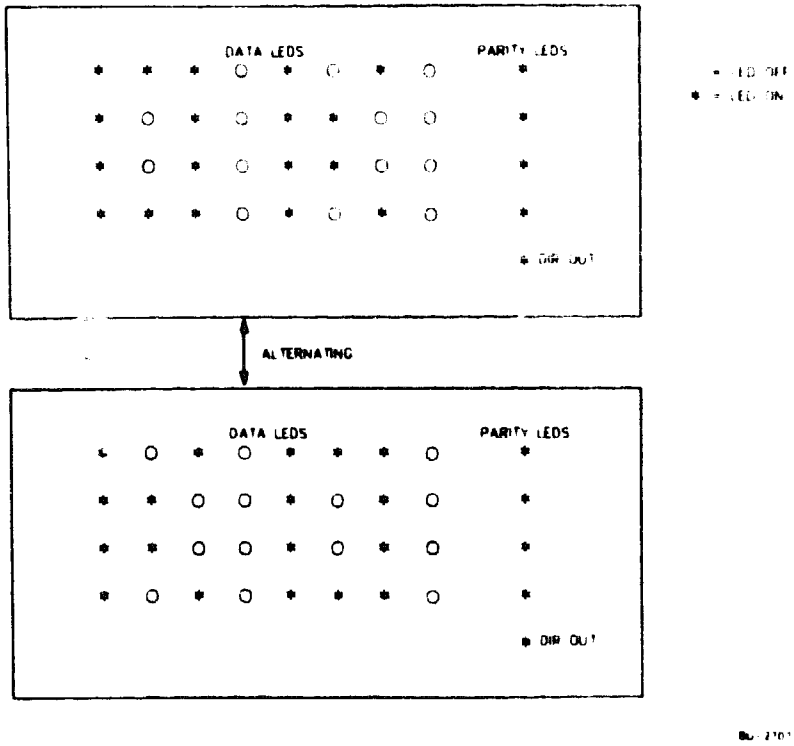


Figure A-6 Pattern 6

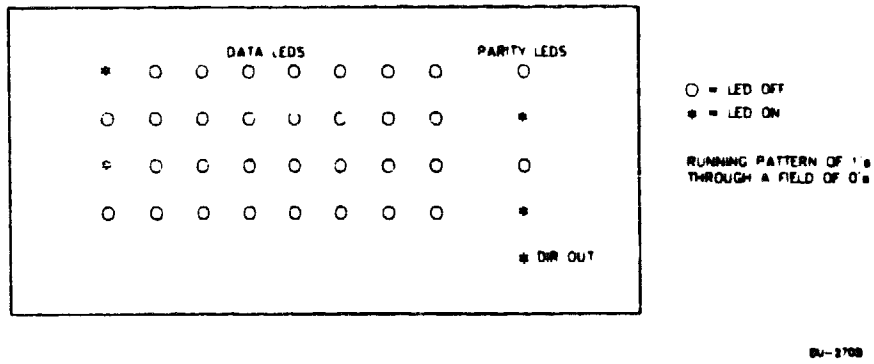


Figure A-7 Parity Test 1

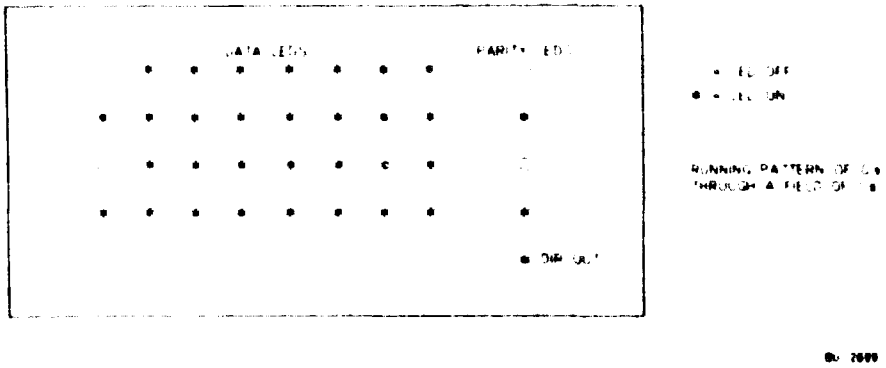
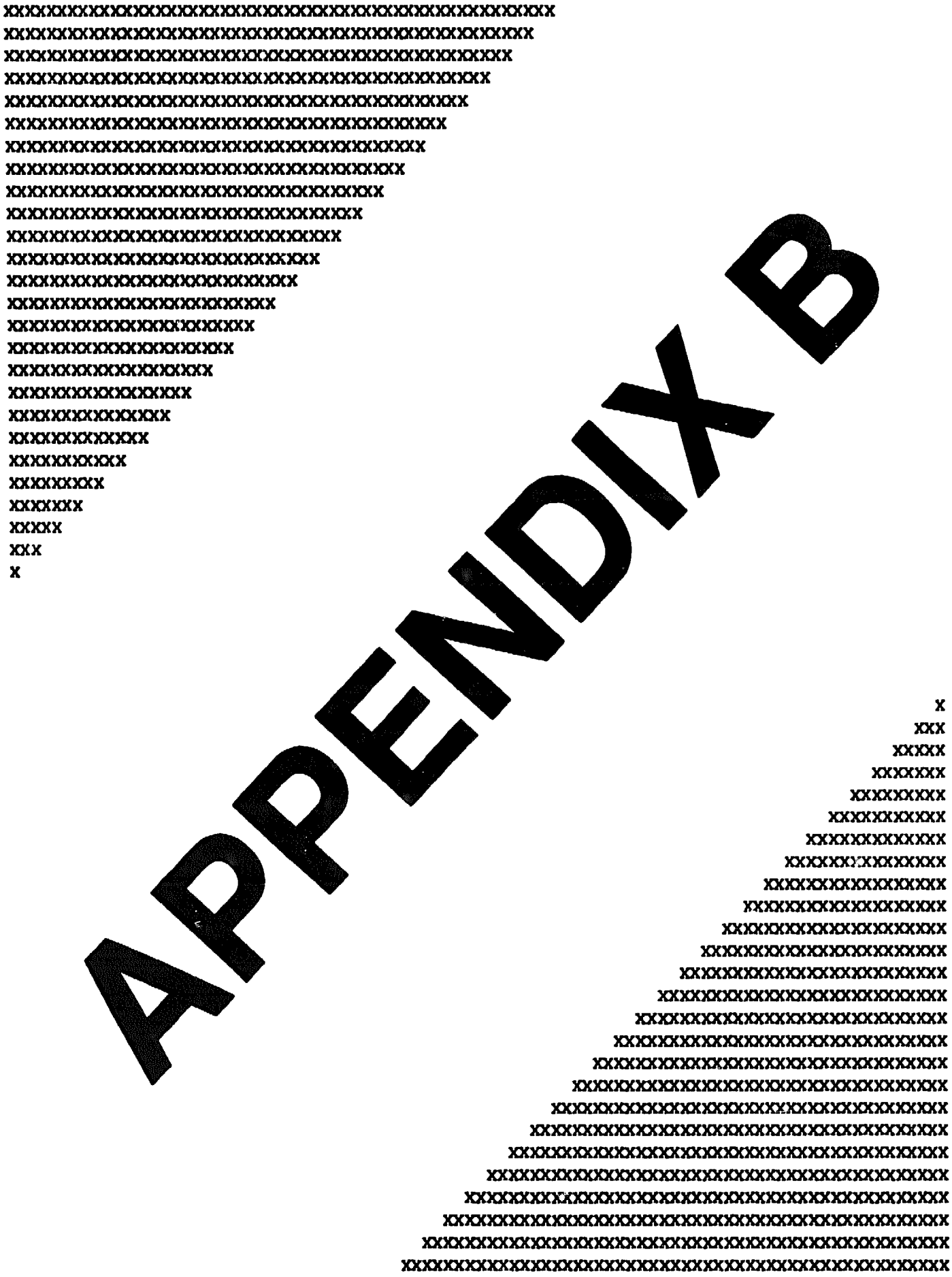


Figure A-8 Parity Test 2

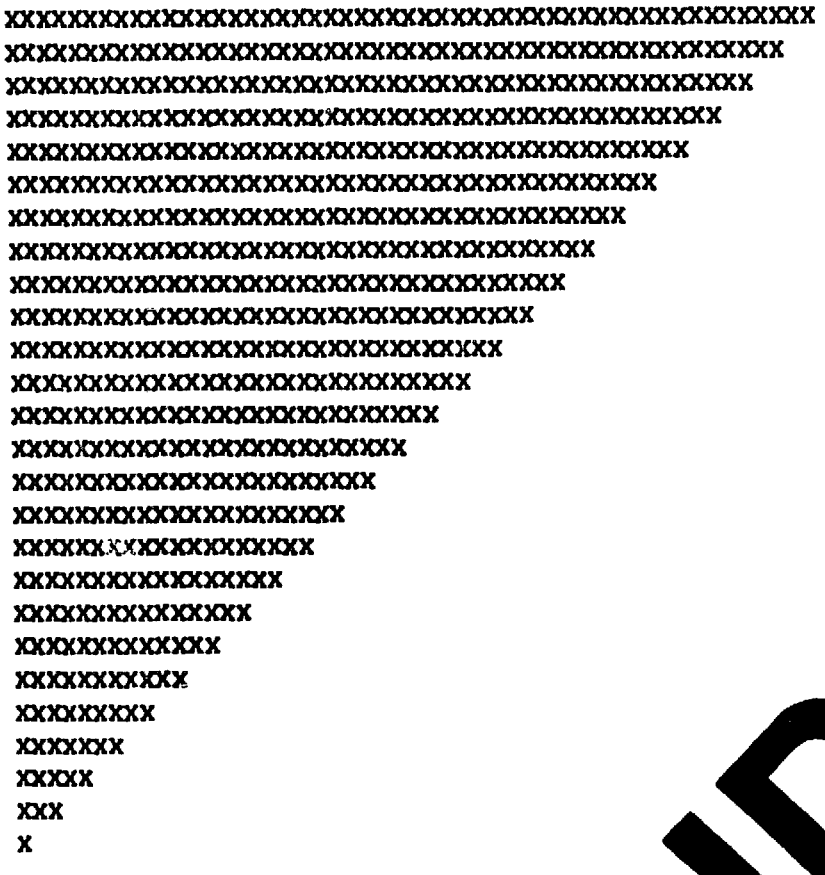

APPENDIX B

B Error Log Format

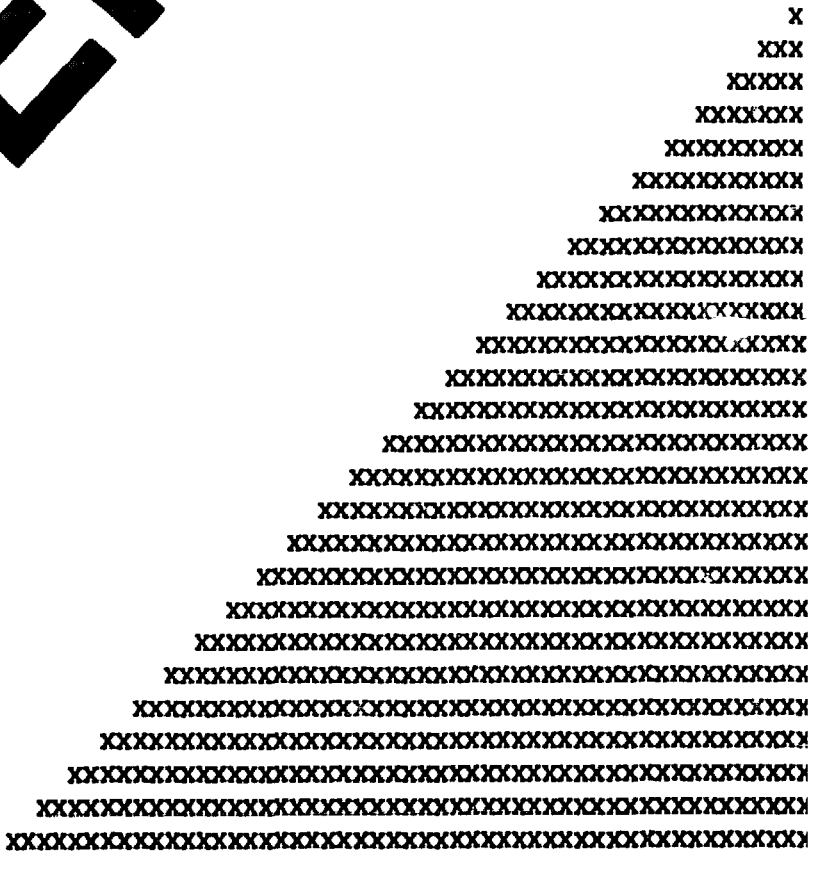
The format for DRB32 Error and Timeout error log entries assumes that the customer is using the VMS Driver that Digital created (UQDRIVER). Customers are encouraged to use and modify the Digital driver, UQDRIVER.EXE, which can be purchased (along with its source code) as a working example of DRB32 use. If the customers use the error log routines from UQDRIVER, then this Error Log Format is used. Digital's driver and error log routine does the following:

- Dumps the BIIC registers
- Dumps the DRB32 required registers
- Provides ten longwords to the customer for logging information of interest to the customer.

Refer to Appendix B of the *Programmer's Manual* for the UQDRIVER Error Log Format.



APPENDIX C



C DRB32 Error Codes

The table below shows the error codes that can appear in the DRB32 ERRREG (bb+168). Comments on certain error codes follow the table.

Table C-1 ERRREG Error Codes

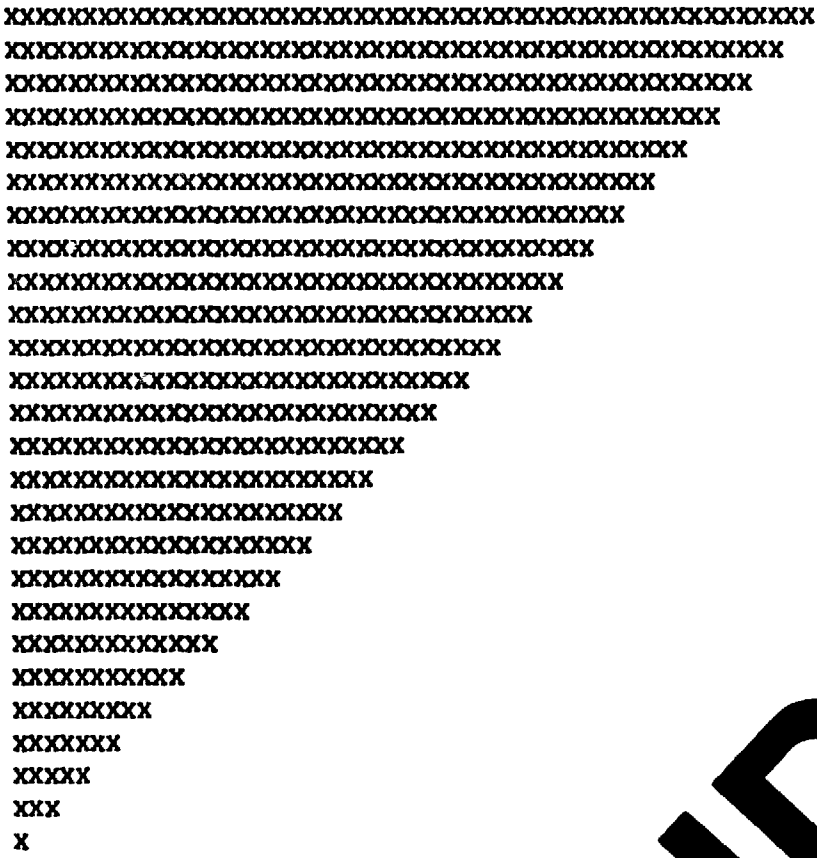
Error Code (Hex)	Meaning
1	Error during DMA transaction
9	Parity error on first map register or CURBOFF read
A	Parity error on map register read
B	Top of map table reached, and byte count not equal 0 (i.e., out of map registers)
11	Abort occurred during DMA transfer
12	VAXBI stop command received
13	Illegal microcode address (microcode missing)
14	VAXBI Slave decode error (i.e., a VAXBI transaction that the microcode does not support was received)

These values appear in the Error Register in the DRB32 address space. The values are valid only when the ERROR bit in the DRB32 Interrupt Flag Register is set.

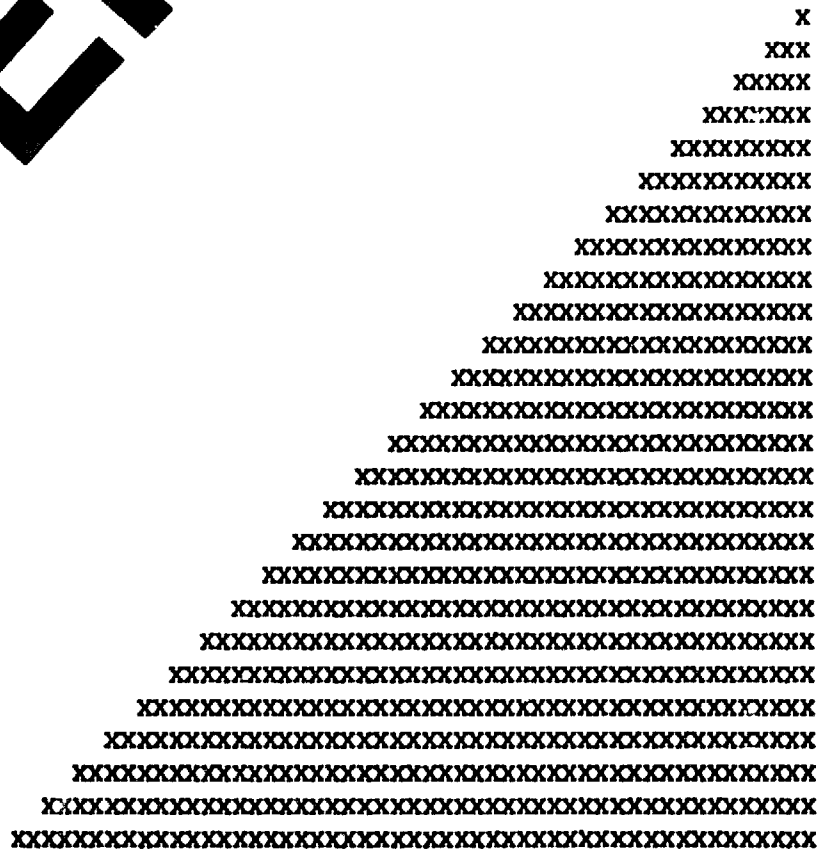
"Error during DMA transaction" (Error 1) means a VAXBI error occurred while the DRB32 microcode was performing a VAXBI transaction. To find more detail, look in the BIIC bus error register (bb+8) for both the master node (the DRB32) and the slave node (generally the memory). Some "error during DMA transaction" errors may be caused by parity errors. Parity errors are indicated by two bits in the Parallel Port Setup and Test Register and might be caused by incorrect user-supplied parity.

The parity errors (Errors 9 and A) on first map register read or subsequent map register read mean either that data was stored in the map registers with bad parity, or the map register location is faulty. There are two possibilities with these errors because the microcode fetches map registers in two different routines.

1. When a DMA transaction is starting, the microcode is generating the first physical address from the map register and from the contents of CURBOFF.
2. The microcode routine that deals with page crossings fetches the next map register contents, and that becomes the next physical address. Top of map register reached and byte count not equal 0 (Error B) is considered an error; it indicates that either the map register pointers in the hardware are faulty, or a programming error was made (the programmer didn't allocate and set up enough map registers to map a buffer of the size indicated by the byte count register).



APPENDIX D



D DRB32 Self-Test Error Codes

Table D-1 lists the self-test error codes.

Table D-1 Self-Test Error Codes

Test Number (Hex)	Error Code (Hex)	Meaning
1	1	Register Selection Failure
	2	Condition Code Failure
	3	Addressing Mode 1 Failure
	4	Addressing Mode 2 Failure
	5	Addressing Mode 3 Failure
	6	Addressing Mode 4 Failure
	7	Addressing Mode 5 Failure
	8	Addressing Mode 6 Failure
	9	Addressing Mode 7 Failure
	A	JSR or RTS Instruction Failed
	B	ROLB Instruction Failed
	C	RORB Instruction Failed
	D	ROL Instruction Failed
	E	ROR Instruction Failed
	F	ABCD Instruction Failed
	10	SBCB Instruction Failed
	11	ADC Instruction Failed

Table D-1 Self-Test Error Codes (Continued)

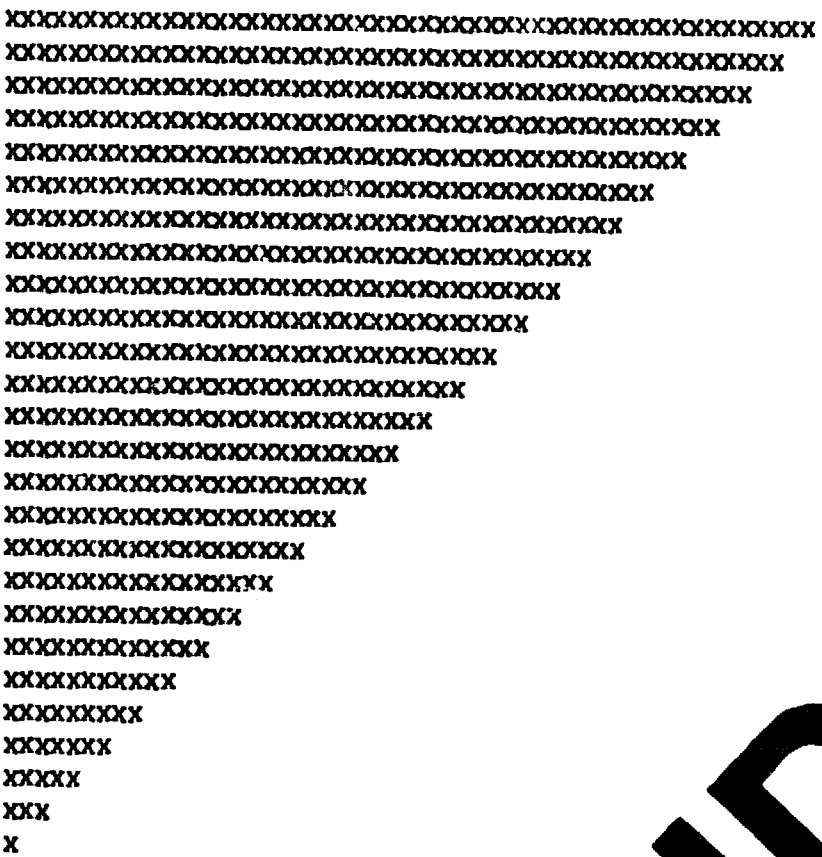
Test Number (Hex)	Error Code (Hex)	Meaning
	12	SBC Instruction Failed
	13	ADD Instruction Failed
	14	SUB Instruction Failed
2	1	Low Byte Error
	2	High Byte Error
3	1	T-11 RAM Error
4	1	CB RAM Error
5	1	Map table Error
6	1	BCAI RAM Error
7	1	BICSR - VAXBI Read Error
	2	BICSR - EV Code Error
	3	BICC Failed Self-Test
	4	BCICSR - VAXBI Write Error
	5	BCISCR - EV Code Error
	6	CB RAM (No T-11 Access) - VAXBI Write Error
	7	CB RAM (No T-11 Access) - EV Code Error
	8	CB RAM (No T-11 Access) - VAXBI Read Error
	9	CB RAM (No T-11 Access) - EV Code Error
	A	CB RAM (No T-11 Access) - Data Error
8	1	CB RAM (No T-11 Access) - VAXBI Write Error
	2	CB RAM (No T-11 Access) - EV Code Error
	3	CB RAM (No T-11 Access) - VAXBI Read Error
	4	CB RAM (No T-11 Access) - EV Code Error
	5	CB RAM - Data Error
9	1	CB RAM (No T-11 Access) - VAXBI Write Error
	2	CB RAM (No T-11 Access) - EV Code Error
	3	CB RAM (T-11 Access) - VAXBI Read Error

Table D-1 Self-Test Error Codes (Continued)

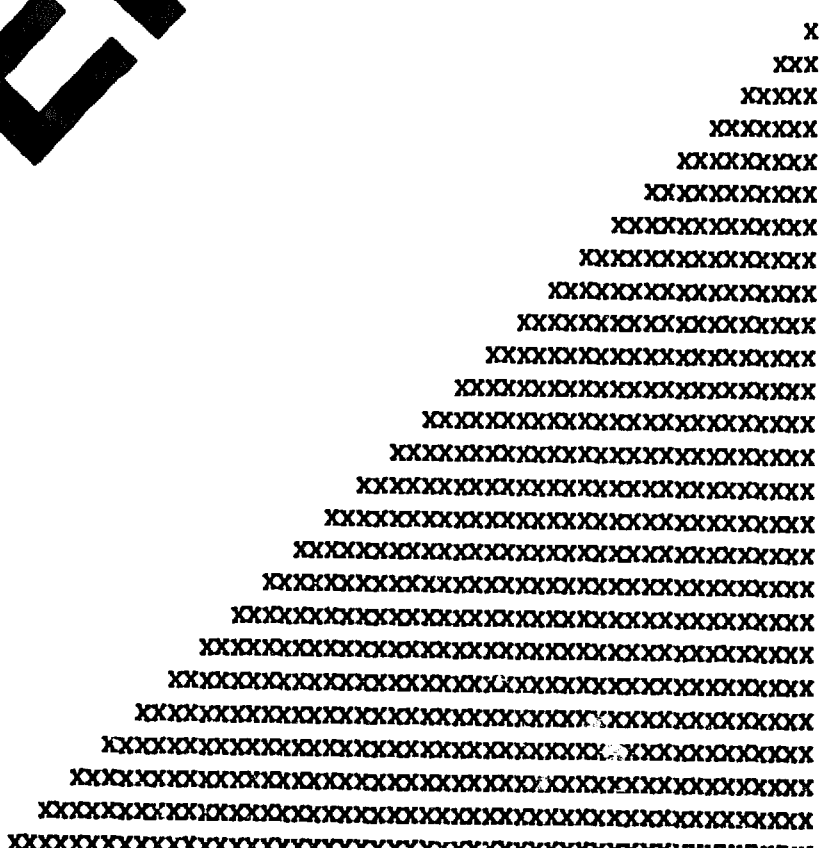
Test Number (Hex)	Error Code (Hex)	Meaning
	4	CB RAM (T-11 Access) - EV Code Error
	5	BOFF - VAXBI Write Error
	6	BOFF - EV Code Error
	7	Bad Byte Offset - Physical Address Reg Read
	8	No T-11 Interrupt
	9	DMA Port - VAXBI Read Error
	A	DMA Port - Data Error
	B	Bad map (15:9) - Physical Address Reg Read
	C	Bad Map (29:9) or Bad LEN - Phys Adr Reg Read
	D	IODAT - VAXBI Write Error
	E	IODAT - EV Code Error
	F	No T-11 Interrupt
	10	DMA Port - VAXBI Write Error
	11	CB RAM (No T-11 Access) - VAXBI Read Error
	12	CB RAM (No T-11 Access) - EV Code Error
	13	DMA Port - Data Error
A	1	CB RAM (No T-11 Access) - VAXBI Write Error
	2	CB RAM (No T-11 Access) - EV Code Error
	3	CB RAM (T-11 Access) - VAXBI Read Error
	4	CB RAM (T-11 Access) - EV Code Error
	5	CURBOFF - VAXBI Write Error
	6	CURBOFF - EV Code Error
	7	No T-11 Interrupt
	8	No Error Interrupt
	9	Wrong Error Code
B	1	CB RAM (T-11 Access) - VAXBI Write Error
	2	CB RAM (T-11 Access) - EV Code Error

Table D-1 Self-Test Error Codes (Continued)

Test Number (Hex)	Error Code (Hex)	Meaning
	3	CB RAM (T-11 Access) - VAXBI Write Error
	4	CB RAM (T-11 Access) - EV Code Error
	5	No Parity Error
	6	BER - VAXBI Write Error
	7	BER - EV Code Error
C	1	No T-11 Interrupt
D	1	No T-11 Interrupt
	2	No COS Interrupt
E	1	DRBIRL Register - Access Error
	2	RXCD Register - Access Error



APPENDIX E



E DRB32 Registers

The following sections describe the registers of the DRB32. These include:

- BIIC registers
- DRB32 module registers

In the lists below, "bb" is the base address of the DRB32 adapter node space on the VAXBI bus.

E.1 BIIC Registers

The BIIC registers include the five VAXBI required registers and the thirteen BIIC specific device registers. These registers are shown in the following tables. Note that the T-11 does not have direct access to these registers (but can do VAXBI loopback transactions). For more detailed information on BIIC registers, see the *VAXBI Options Handbook*.

Table E-1 BIIC Registers

VAXBI Address	Register Name	
bb+00	DTYPE	Device Register
bb+04	BICSR	VAXBI Control and Status Reg
bb+08	BER	Bus Error Register
bb+0C	EINRECSR	Error Interrupt Control Reg
bb+10	INTRDES	Interrupt Destination Reg
bb+14	IPINTRMSK	IPINTR Mask Reg
bb+18	FIPSDES	Force IPINTR/STOP Dest Reg
bb+1C	IPINTRSRC	IPINTR Source Reg

Table E-1 BIIC Registers (Continued)

VAXBI Address	Register Name	
bb+20	SADR	Starting Address Reg
bb+24	EADR	Ending Address Reg
bb+28	BCICSR	BCI Control Reg
bb+2C	WSTAT	Write Status Reg
bb+30	FIPSCMD	Force IPINTR/STOP Command Reg
bb+40	UINTRCSR	User Interface Interrupt Control Reg
bb+F0	GPR0	General Purpose Register 0 (Self-Test Error Code)
bb+F4	GPR1	General Purpose Register 1 (DRB32 Self-Test and Firmware Rev. Level)
bb+F8	GPR2	General Purpose Register 2
bb+FC	GPR3	General Purpose Register 3

E.2 DRB32 Module Registers

The DRB32 module registers include the following:

- Parallel I/O port registers
- T-11 registers

E.3 Parallel Port Registers

The DRB32 parallel I/O port registers are shown in the following list.

Table E-2 DRB32 Parallel I/O Port Registers

VAXBI Address	Register Name	
bb+015C	BAR	BCAI Access Reg
bb+0160	PARCSR	Par Port Contrl and Stat Reg
bb+0164	PARSTS	Par Port Setup and Test Reg
bb+0168	ERRREG	Error Reg

Table E-2 DRB32 Parallel I/O Port Registers (Continued)

VAXBI Address	Register Name	
bb+016C	DRBIE	DRB32 Inter Enable Reg
bb+0170	DRBIRL	DRB32 Inter Request Lev Reg
bb+0174	DRBIFR	DRB32 Inter Flag Reg
bb+0178	CURMR	Curr Map Reg Pointer
bb+017C	CURTOMR	Curr Top Of Map Reg Area
bb+0180	CURBOFF	Curr Address Byte Offset
bb+0184	CURBLFT	Curr No. of Bytes in Seg
bb+0188	PHYSADDR	Physical Address
bb+018C	IODAT	I/O Data Reg
bb+0190	IOCTL	I/O Control Reg
bb+0194	NXTMR	Next Map Reg Pointer
bb+0198	NXTTOMR	Next Top of Map Reg Area
bb+019C	NXTBOFF	Next Address Byte Offset
bb+01A0	NXTBLFT	Next Bytes Left in Seg
bb+01A4	T11ICR	RESERVED
bb+01A8	T11MDTR	RESERVED
bb+01AC	DPWR	Data Port Width Reg
bb+01B0	BOBP	RESERVED

The DRB32 parallel I/O port registers are explained briefly below.

BAR — BCAI Access Register is used to read the registers in the BCAI (which are not directly accessible to the VAXBI). This register is normally used for extracting stranded data from the dual-octaword register after a DRB32 error condition (or the setting of the ABORT bit in the PARCSR) halts a DMA operation in which data is being read from a user device.

PARCSR — The Parallel Port Control and Status Register is used by a driver running in the VAX processor to start or abort a data transfer. The register also provides information about the current status of the transfer.

PARSTS — The Parallel Port Setup and Test Register is loaded by the VAXBI with information about the configuration and format of the current data transfer. This register also has diagnostic functions.

ERRREG — The Error Register is loaded by the DRB32 microcontroller whenever an error condition occurs. These bits are cleared when a new data transfer is started, by setting the GO bit in the Parallel Port Control Register (PARCSR). Refer to Appendix B.

DRBIE — The DRB32 Interrupt Enable Register controls which events allow the DRB32 to send an interrupt to a VAXBI node.

DRBIRL — The DRB32 Interrupt Request Level Register is written with the DRB32 interrupt priority level.

DRBIFR — The DRB32 Interrupt Flag Register indicates which events caused the DRB32 to issue an interrupt transaction to a VAXBI node.

CURMR — The Current Map Register Pointer contains the Central Bus RAM (CB RAM) address of the map register being used for the physical address of the first page of the current data transfer. It is loaded at the start of a data transfer with the address of the bottom of the map register table.

CURTOMR — The Current Top of Map Register Area Pointer contains the location of the top of the current map table in the CB RAM.

CURBOFF — The Current Byte Offset Register is concatenated on the bottom of the map register pointed to by the CURMR, to form the physical address of the start of the buffer. This register is loaded with the offset from the start of a page to the first byte of the buffer on that page.

CURBLFT — Loaded at start of a transfer with number of bytes to be transferred. If transfer aborts, CURBLFT contains the number of bytes NOT transferred.

(WRITE) The Current Number of Bytes Left in Segment Register is loaded by the VAXBI with the number of bytes to be transferred in the current segment transfer.

(READ) Reading this register indicates the number of bytes in the current segment left to be transferred to the user device or the VAXBI.

PHYSADDR — The Physical Address Register is the physical address (in VAXBI address space) of the start of the current page in the segment transfer. Generated by microcontroller (by concatenating first map register contents with CURBOFF).

IODAT — The I/O Data Register contains the current state of the parallel port data lines.

IOCTL — The I/O Control Register reflects the value of the inputs and outputs of the DRB32 control bus, which can be used to communicate control and/or protocol information to and from the user devices.

The next four registers are used to describe a second buffer if double buffering is used. When double buffering is used, these registers are loaded into CURxxx when the current transfer is completed. The contents of the NXTxxx registers are similar to the contents of the corresponding CURxxx registers.

NXTMR — The Next Map Register Pointer's contents are loaded into the Current Map Register Pointer at the end of the segment transfer.

NXTTOMR — The Next Top of the Register Area Register's contents are loaded into the Top of the Register Area Register at the end of the current segment transfer.

NXTBOFF — The Next Address Byte Offset Register's contents are loaded into the Current Byte Offset Register at the end of the current segment transfer.

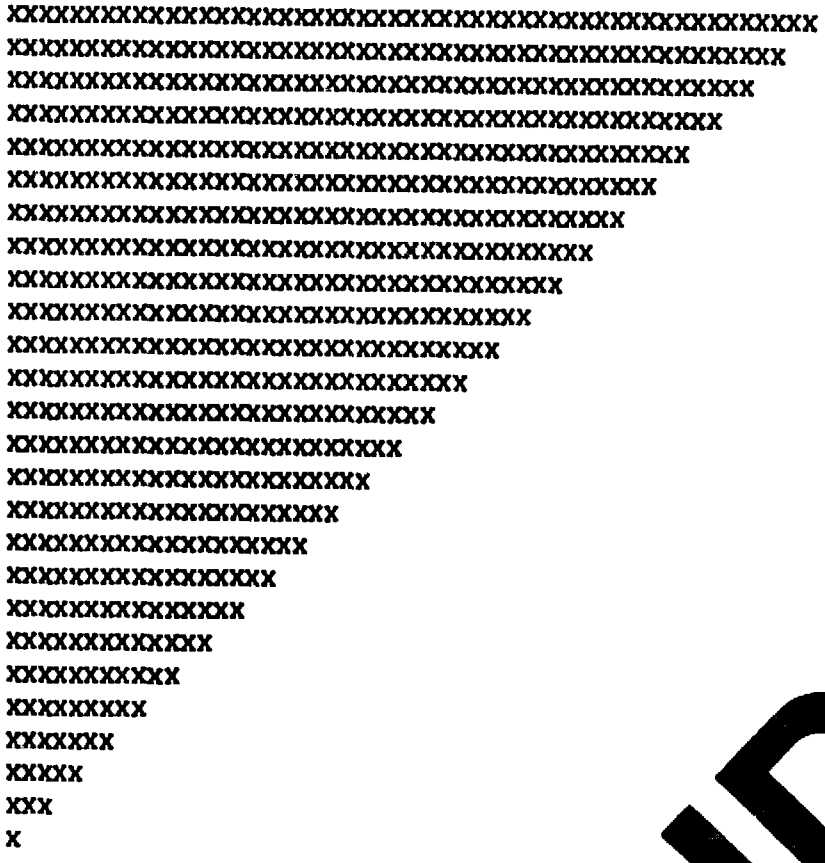
NXTBLFT — The Next Number of Bytes Left in Segment Register's contents are loaded into the Current Number of Bytes Left in Segment Register at the end of the current segment transfer.

T11ICR — RESERVED

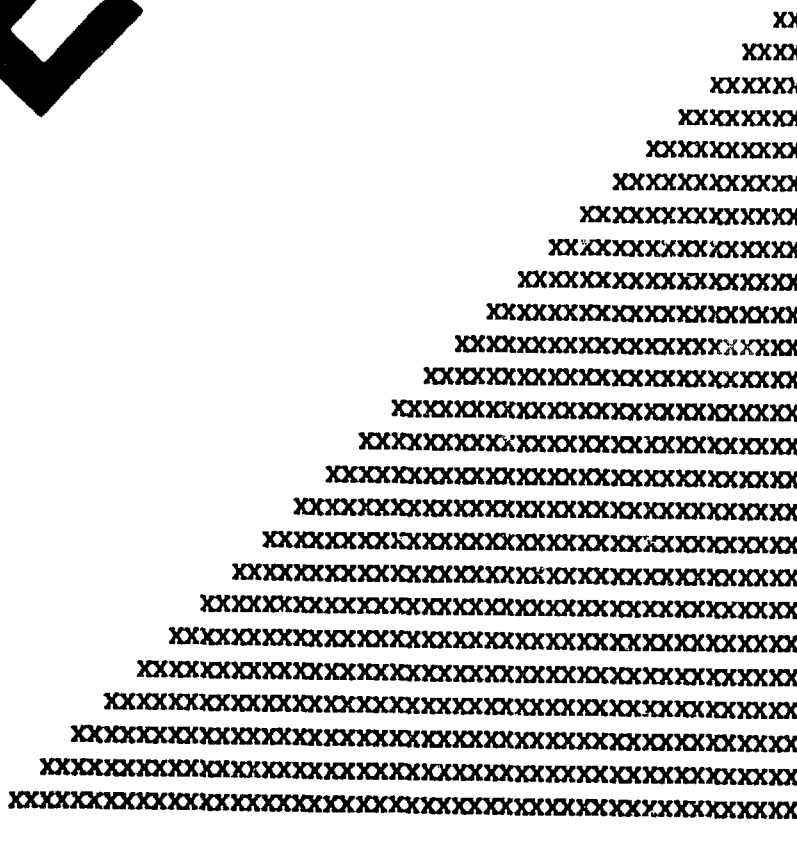
T11MDTR — RESERVED

DPWR — The Data Port Width Register determines whether data is transferred in 8-, 16-, or 32-bits through the user port.

BOBP — RESERVED



APPENDIX F



F Glossary

This glossary describes the DRB32.

Adapter A node that interfaces other buses, communication lines, or peripheral devices to the VAXBI bus.

Arbitration Cycle A cycle during which nodes arbitrate for control of the VAXBI bus.

Assert To cause a signal to take the "true" or asserted state.

Asserted To be in the "true" state.

Assertion The transition of a signal from deasserted to asserted.

Atomic Pertaining to an indivisible operation.

Bandwidth The data transfer rate measured in information units transferred per unit of time (for example, megabytes per second). All bandwidth figures quoted in this manual take into account command/address and embedded ARB cycle overhead.

BCI VAXBI chip interface; synchronous interface bus that provides for all communication between the BIIC and the user interface.

BIIC Backplane interconnect interface chip; chip that serves as a general purpose interface to the VAXBI bus.

BIIC CSR Space The first 256 bytes of the 8-KB node space, which is allocated to the BIIC's internal registers. See also Node Space.

BIIC-Generated Request A transaction request generated by the BIIC rather than by the user interface. The BIIC can request only error interrupts.

BIIC-Generated Transaction A transaction performed solely by the BIIC with no assistance from the master port interface. Only INTR and IPINTR transactions can be independently generated by the BIIC. The user interface initiates BIIC-generated transactions by using the IPINTR/STOP force bit, the user interface or error interrupt force bits, or by asserting one of the BCI INT<7:4> L lines. A BIIC-generated transaction can also result from a BIIC-generated request, which results from a bus error that sets a bit in the Bus Error Register.

Block Mode DMA data transfer mode.

Bus Access Latency The delay from the time a node desires to perform a transaction on the VAXBI bus until it becomes master.

Bus Adapter A node that interfaces the VAXBI bus to another bus.

Busy Extension Cycle A bus cycle during which a VAXBI node not necessarily the master or slave of the transaction, asserts the VAXBI BSY L line to delay the start of the next transaction.

CB RAM Central Bus RAM. Part of the DRB32 parallel I/O port; contains map registers and 8-KB storage.

Client User program that links with DRB32$MESSAGE subroutine package to communicate with another user program.

Command/Address Cycle The first cycle of a VAXBI transaction. The information transmitted in this cycle is used to determine slave selection. In some cases, the data on the VAXBI D<31:0> lines is not an actual address, but it serves the same purpose: to select the desired slave nodes(s).

Command Confirmation The response sent by the slave(s) to the bus master to confirm participation in the transaction.

Command Confirmation Cycle The third cycle in a VAXBI transaction during which slave(s) confirm participation in the transaction.

Configuration Data Data loaded into the BIIC on power-up that includes the device type and revision code, the parity mode, and the node ID.

Cycle The basic bus cycle of 200 nanoseconds (nominal), which is the time it takes to transfer the smallest piece of information on the VAXBI bus. A cycle begins at the leading edge of T0/50 and continues until the leading edge of the next T0/50.

Data Cycle A cycle in which the VAXBI data path is dedicated to transferring data (such as read or write data, as opposed to command/address or arbitration information) between the master and slave(s).

Data Mode Programmed I/O data transfer mode.

Data Transfer Commands VAXBI commands that involve the transfer of data as well as command/address information: read-type, write-type, IDENT, and BDCST commands.

Deassert To cause a signal to be in the "false" or deasserted state.

Deassertion The transition of a signal from asserted to deasserted.

Decoded ID The node ID expressed as a single bit in a 16-bit field.

Device Type A 16-bit code that identifies the node type. This code is contained in the BIIC's Device Register.

DMA Adapter An adapter that directly performs block transfers of data to and from memory.

DMA Mode Data transfer mode directly to and from memory; block mode.

DRB32-E The external driver option of the DRB32 adapter, consisting of the T1022 module, the T1024 module, and connecting cabling.

DRB32-M The basic DRB32 module option (T1022).

DRB32-W The DR11-W interface option of the DRB32 adapter, consisting of the T1022 and T1023 boards and connecting cabling.

DRB32$MESSAGE Program consisting of subroutines that enable communication between two DRB32 adapters.

DRB32$QIO Program for the DRB32-M/-E, consisting of subroutines that check UQDRIVER; they are examples of how to access device driver from specific program environments.

DRB32$WQIO Program for the DRB32-W, consisting of subroutines that check UQWDRIVER; they are examples of how to access device driver from specific program environments.

Encoded ID The node ID expressed as a 4-bit binary number. The encoded ID is used for the master's ID transmitted during an embedded ARB cycle.

EVDRH The DRB32 level 3 diagnostic.

H Designates a high-voltage logic level (that is, the logic level closest to Vcc). Contrast with L.

Interlock Commands The two commands, IRCI (Interlock Read with Cache Intent) and UWMCI (Unlock Write Mask with Cache Intent), that are used to implement atomic read-modify-write operations.

Internode Transfer A VAXBI transaction in which the master and slave(s) are in different VAXBI nodes. Contrast with Intranode Transfer.

Interrupt Port Those BCI signals that are used in generating INTR transactions.

Interrupt Port Interface That portion of user logic used to interface to the interrupt port of the BIIC.

Interrupt Vector In VAX/VMS systems, an unsigned binary number used as an offset into the system control block. The system control block entry pointed to by the VAXBI interrupt vector contains the starting address of an interrupt handling routine. (The system control block is defined in the *VAX-11 Architecture Reference Manual.*)

Intranode Transfer A transaction in which the master and slave are in the same node. Loopback transactions are intranode transfers. Contrast with Internode Transfer.

L Designates a low-voltage logic level (that is, the logic level closest to ground). Contrast with H.

Local Memory VAXBI memory that can be accessed without using VAXBI transactions; for example, VAXBI-accessible memory on a single board computer.

Loopback Extension Cycle A cycle of a loopback transactions during which a node asserts both BI BSY L and BI NO ARB L to delay the start of the next transaction.

Loopback Mode Diagnostic mode in which DRB32 does loopback transactions instead of VAXBI transactions.

Loopback Request A request from the master port interface asserted on the BCI RQ<1:0> L lines that permits intranode transfers to be performed without using the VAXBI bus.

Loopback Transaction A transaction in which information is transferred within a given node without use of the VAXBI data path. Contrast with VAXBI Transaction.

Mapped Adapter A DMA adapter that performs data transfers between a system with a contiguous memory space and VAXBI address space (in which memory need not be contiguous). The mapping is done by using a set of map registers located in the adapter.

Master The node that gains control of the VAXBI bus and initiates a VAXBI or loopback transaction. See also Pending Master.

Master Port Those BCI signals used to generate VAXBI or loopback transactions.

Master Port Interface That portion of user logic that interfaces to the master port of the BIIC.

Master Port Request A request (either VAXBI or loopback) generated by the master port interface through the use of the BCI RQ<1:0> L lines.

Master Port Transaction Any transaction initiated as a result of a master port request.

Module A single VAXBI card that attaches to a single VAXBI connector.

Multiresponder Commands VAXBI commands that allow for more than one responder. These include the INTR, IPINTR, STOP, INVAL, and BDCST commands.

Node A VAXBI interface that occupies one of 16 logical locations on a VAXBI bus. A VAXBI node consists of one or more VAXBI modules.

Node ID A number that identifies a VAXBI node. The source of the node ID is an ID plug attached to the backplane.

Node Reset A sequence that causes an individual node to be initialized; it is initiated by the setting of the Node Reset bit in the VAXBI Control and Status Register.

Node Space An 8-KB block of I/O addresses that is allocated to each node. Each node has a unique node space based on its node ID.

Odd Parity The parity line is asserted if the number of asserted lines in the data field is an even number. The DRB32 uses odd parity.

Parity Mode Specifies whether parity is generated by the BIIC or by the User interface.

Pending Master A node that has won an arbitration but has not yet begun a transaction.

Pending Request A request of any type, whether from the master port or a BIIC-generated request, that has not yet resulted in a transaction.

Pipeline Request A request from the master port that is asserted prior to the deassertion of BCI RAK L for the present master port transaction; that is, a new request is posted prior to the completion of the previous transaction.

Power-Down/Power-Up Sequence The sequencing of the BI AC LO L and BI DC LO L lines upon the loss and restoration of power to a VAXBI system. See also System Reset.

Private Memory Memory that cannot be accessed from the VAXBI bus.

Programmed I/O (PIO) Adapter An adapter that does not access memory on the VAXBI bus but interacts only with a host processor.

Programmed I/O Mode Data transfer mode in which the processor must intervene on each Read or Write transaction; data mode.

Read Access Time The delay from the time a node requests read data until it receives that data from the VAXBI bus.

RCLK (Receive Clock) The clock phase during which information is received from the VAXBI bus; equivalent to T100/150.

Read Data Cycle A data cycle in which data is transmitted from a slave to a master.

Read-Type Commands Any of the various VAXBI read commands, including READ, RCI (Read With Cache Intent), and IRCI (Interlock Read With Cache Intent).

RESERVED Code A code reserved for use by Digital.

RESERVED Field A field reserved for use by Digital. The node driving the bus must ensure that all VAXBI lines in the RESERVED field are deasserted.

Reset Module In a VAXBI system, the logic that monitors the BI RESET L line and any battery back-up voltages and that initiate the system reset sequence.

Resetting Node The node that asserts the BI RESET L line.

Retry State A state that the BIIC enters upon receipt of a RETRY confirmation code from a slave. If the master reasserts the transaction request, the BIIC resends the transaction without having the user interface provide the transaction information again. The command/address information and the first data longword, if a write transaction, are stored in buffers in the BIIC.

Single-Responder Commands VAXBI commands that allow for only one responder. These include read- and write-type commands and the IDENT command. Although multiple nodes can be selected by an IDENT, only one returns a vector.

Slave A node that responds to a transaction initiated by a node that has gained control of the VAXBI bus (the master).

Slave Port Those BCI signals used to respond to VAXBI and loopback transactions.

Slave Port Interface That portion of user logic that interfaces to the slave port of the BIIC.

STALL Data Cycle A data cycle of a read- or write-type transaction during which the slave asserts the STALL CNF code to delay the transmission of the next data word.

System Reset An emulation of the power-down/power-up sequence that causes all nodes to initialize themselves; initiated by the assertion of the BI RESET L line.

T-11 16-Bit PDP-11 microprocessor.

Target Bus The bus that a VAXBI node interfaces to the VAXBI bus.

Transaction The execution of a VAXBI command. The term "transaction" includes both VAXBI and loopback transactions.

UQDRIVER The VMS device driver for the DRB32-M/-E that is supplied with the DRB32 software kit.

UQWDRIVER The VMS device driver for the DRB32-W that is supplied with the DRB32 software kit.

User Interface All node logic exclusive of the BIIC.

User Interface CSR Space That portion of each node space allocated for user interface registers. The user interface CSR space is the 8-KB node space minus the lowest 256 bytes, which compose the BIIC CSR space.

User Interface Request A transaction request from the user interface, which can take the form of a master port request, an assertion of a BCI INT<7:4> L line, or the setting of a force bit.

User Port Loopback Mode DRB32 diagnostic mode in which a longword of data written to either IODAT or IOCTL registers can be read back from those registers, without sending data out on the bus. This mode is entered by setting the LB bit in the PARSTR.

VAX Interrupt Priority Level (IPL) In VAX/VMS systems, any number between 0 and 31 that indicates the priority level of an interrupt with 31 being the highest priority. When a processor is executing at a particular level, it accepts only interrupts at a higher level; and on acceptance starts executing at that higher level.

VAXBI Corner The physical corner of a board that connects to the VAXBI; must include the BIIC chip and other circuitry.

VAXBI Loopback Mode DRB32 diagnostic mode in which all VAXBI transactions issued by the DRB32 are VAXBI loopback transactions to the DRB32 slave port. This mode is entered by the VAXBI or the T-11 setting the BILB bit in the PARSTR.

VAXBI Primary Interface The portion of a node that provides the electrical connection to the VAXBI signal lines and implements the VAXBI protocol; for the DRB32, this is the BIIC.

VAXBI Request A request for a VAXBI transaction from the master port interface that is asserted on the BCI RQ<1:0> L lines.

VAXBI System All VAXBI cages, VAXBI modules, reset modules, and power supplies that are required to operate a VAXBI bus. A VAXBI system can be a subsystem of a larger computer system.

VAXBI Transaction A transaction in which information is transmitted on the VAXBI signal lines. Contrast with Loopback Transaction.

Vector Data Cycle A data cycle in which an interrupt vector is transmitted from a slave to a master.

Window Adapter A bus adapter that maps addresses that fall within one contiguous region (a "window") of a bus's address space into addresses in a window (possibly in a different region) in another bus's address space.

Window Space A 256-KB block of I/O addresses allocated to each node based on node ID and used by bus adapters to map VAXBI transactions to other buses.

Write-Type Commands Any of the various VAXBI write commands, including WRITE. WCI (write with cache intent), WMCI (write mask with cache intent), and UWMCI (unlock write mask with cache intent).

Write Data Cycle A data cycle in which data is transmitted from a master to a slave.

XADRIVER DR11-W device driver supplied with VMS.


INDEX

Index