KA660 CPU System Maintenance

Order Number EK–398AA–MM–001

Digital Equipment Corporation
Maynard, Massachusetts

First Printing, December 1990

The Reader's Comments form at the end of this document requests your critical evaluation to assist in preparing future documentation.

FCC NOTICE: The equipment described in this manual generates, uses, and may emit radio frequency energy. The equipment has been type tested and found to comply with the limits for a Class A computing device pursuant to Subpart J of Part 15 of FCC Rules, which are designed to provide reasonable protection against such radio frequency interference when operated in a commercial environment. Operation of this equipment in a residential area may cause interference, in which case the user at his own expense may be required to take measures to correct the interference.

S1599

This document was prepared using VAX DOCUMENT, Version 1.2.

Contents

Chapter 3 KA660 Firmware

Chapter 4 Troubleshooting and Diagnostics

Appendix A KA660 CPU Address Assignments

Appendix B Programming Parameters for RF-Series ISEs

Index

Examples

Figures

Tables

Preface

This guide describes the base system, configuration, ROM-based diagnostics, and troubleshooting procedures for systems containing the KA660 CPU.

Intended Audience

This guide is intended for use by Digital Customer Services personnel and qualified self-maintenance customers.

Organization

This guide has four chapters and two appendixes, as follows:

Chapter 1 describes the KA660/MS650–Bn CPU and memory subsystem, and the RF-series Integrated Storage Elements (ISEs).

Chapter 2 contains system configuration guidelines, and provides a table listing current, power, and bus loads for supported options. It also describes the Digital Storage Systems Interconnect (DSSI) bus interface cabling between the RF-series ISEs, CPU, the CPU I/O panel, and the system control panel (SCP). (The control panel is known as the SCP on the BA430 enclosure and as the operator control panel (OCP) on the BA215 enclosure.)

Chapter 3 describes the firmware that resides in ROM on the KA660, and provides a list of console error messages and their meaning.

Chapter 4 describes the KA660 diagnostics and the diagnostics that reside on the RF-series ISEs.

Appendix A lists the KA660 address space.

Appendix B describes procedures for setting parameters on an ISE.

Conventions

The following conventions are used in this manual:

Convention	Meaning
Key	A symbol denoting a terminal key used in text and examples in this book. For example, Break indicates that you press the Break key on your terminal keypad. Return indicates that you press the Return key on your terminal keypad.
Ctrl/C	A symbol indicating that you hold down the Ctrl key while you press the C key.
BOLD	This bold type indicates user input. For example: **>>> BOOT MUA0** This line shows that the user must type BOOT MUA0 at the console prompt.
NOTE	Provides general information about the current topic.
CAUTION	Provides information to prevent damage to equipment or software.
WARNING	Provides information to prevent personal injury.
The following are qualifier and argument conventions:	
[]	an optional qualifier or argument
{}	a required qualifier or argument

Chapter 1
KA660 CPU and Memory Subsystem

1.1 Introduction

This chapter describes the KA660 CPU (Figure 1–1). The KA660 is a quad-height VAX processor module for the Q22-bus (extended LSI–11 bus). It is designed for use in high-speed, real-time applications and for multiuser, multitasking environments. The KA660 employs a cache memory to maximize performance.

There are two variants: the KA660–AA, which runs multiuser software; and the KA660–BA, which runs single-user software.

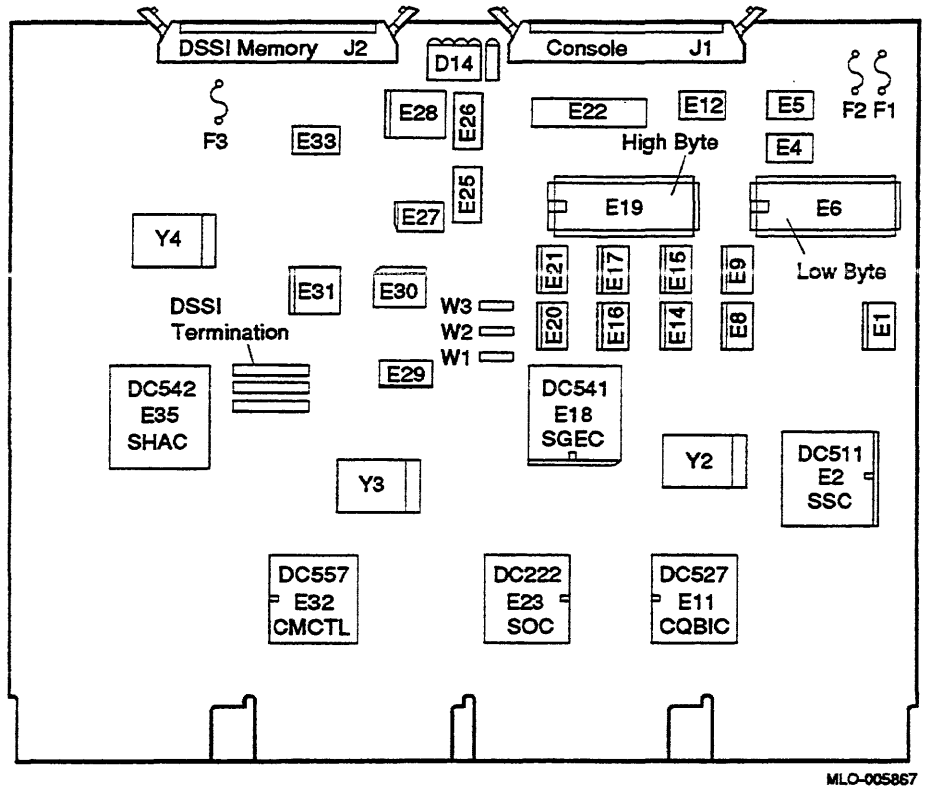
The KA660 is the CPU of the VAX 4000 Model 200, which is housed in either a BA430 or a BA215 enclosure. Refer to the *BA430/BA440 Enclosure Maintenance* manual.

CAUTION: *Static electricity can damage integrated circuits. Always use a grounded wrist strap (PN 29–11762–00) and grounded work surface when working with the internal parts of a computer system.*

The KA660 CPU module and MS650–Bn memory modules combine to form a VAX CPU and memory subsystem that can use the on-board DSSI and Ethernet busses and the Q22-bus to communicate with I/O devices. The KA660 and MS650–Bn modules mount in standard Q22-bus backplane slots that implement the Q22-bus in the AB rows and the CD interconnect in the CD rows. The KA660 can support up to four MS650–Bn modules, if enough Q22/CD slots are available.

The KA660 communicates with the console device through the CPU cover panel (H3602–00), which also contains configuration switches and an LED display. The H3602–00 is described in Section 1.3.

Figure 1–1: KA660 CPU Module



1.2 KA660 Features

The major features of the KA660 CPU are listed below.

- The VAX central processor, which is implemented in a VLSI chip called the SOC, achieves a 35-ns microcycle and a 70-ns bus cycle at an operating frequency of 114 MHz. It supports full VAX memory management with demand paging and a 4-Gbyte virtual address space. The SOC includes a floating-point accelerator with the MicroVAX chip subset of the VAX floating-point instruction set and data types.
- A console port compatible with the VAX processor whose baud rate can be set through an internal switch on the CPU cover panel.

- A set of processor clock registers that support:
  - A VAX standard time-of-year (TOY) clock with support for battery backup. (Batteries are located in the CPU cover panel.)
  - An interval timer with 10-ms interrupts.
  - Two programmable timers, similar in function to the VAX standard interval timer.
- A boot and diagnostic facility with four on-board LEDs. This facility supports an external 4-bit display and configuration switches on the CPU cover panel.
- 256 Kbytes of 16-bit wide ROM.
- A Q22-bus interface.
- A DSSI bus interface.
- An Ethernet interface.

1.2.1 SOC Chip

The SOC chip contains all general purpose registers (GPRs) visible to the VAX processor, several system registers such as CCR, SCBB, the cache memory (6 Kbytes), and all memory management hardware, including a 28-entry translation buffer.

The SOC chip supports the MicroVAX chip subset of the VAX instruction set and data types, plus the following string instructions:

CMPC3
CMPC5
LOCC
MOVC3
MOVC5
SCANC
SKPC
SPANC

The SOC chip provides the following subset of the VAX data types:

Byte
Word
Longword
Quadword
Character string
Variable-length bit field

Support for the remaining VAX data types can be provided through macrocode emulation.

1.2.2 Clock Functions

Clock functions are implemented by the SOC, which includes the clock chip, FPA, CPU, and cache.

- Generates one auxiliary clock for other TTL logic
- Synchronizes reset signal
- Synchronizes data ready and data error signals

1.2.3 Floating-Point Accelerator

The floating-point accelerator is implemented on the SOC chip. The FPA subsystem executes the VAX f_, d_, and g_floating-point instructions (except for CLRx, MOVx, and TSTx), and accelerates the execution of MULL, DIVL, and EMUL integer instructions.

1.2.4 Cache Memory

The KA660 module incorporates a cache memory of six banks to maximize CPU performance. The cache is implemented within the SOC chip. The cache is a 6-Kbyte, six-way associative, write-through cache memory, with a 35-ns cycle time.

1.2.5 Memory Controller

The main memory controller is implemented by a VLSI chip called the CMCTL. The CMCTL contains approximately 25,000 transistors in a 132-pin CERQUAD surface mount package. It supports ECC (error correction code) memory, with a 420-ns cycle time for longword read transfers and a 560-ns cycle time for quadword transfers. It has a 140-ns cycle time for unmasked longword writes and a 490-ns cycle time for masked longword writes.

The maximum amount of main memory supported by KA660 systems is 64 Mbytes on one to four MS650–BA or –BB (16-Mbyte or 8-Mbyte) memory modules, depending on system configuration. The MS650–Bn modules communicate with the KA660 through the MS650–Bn memory interconnect, which utilizes the CD interconnect and a 50-pin ribbon cable.

1.2.6 MicroVAX System Support Functions

System support functions are implemented by the System Support Chip (SSC). The SSC contains approximately 83,000 transistors in an 84-pin CERQUAD surface mount package. The SSC provides console and boot code support functions, operating system support functions, timers, and many extra features, including the following:

- Word-wide ROM unpacking
- 1-Kbyte battery-backed-up RAM
- Halt arbitration logic
- A console serial line
- An interval timer with 10-ms interrupts
- A VAX standard time-of-year (TOY) clock with support for battery backup
- An IORESET register
- Programmable CDAL bus timeout
- Two programmable timers
- A register for controlling the diagnostic LEDs

1.2.7 Resident Firmware

The resident firmware, arranged as words, consists of 256 Kbytes located on two 128-Kbyte x 8-bit wide EPROMs (27010). The firmware gains control when the processor halts, and contains programs that provide the following services:

- Board initialization
- Power-up self-testing of the KA660 and MS650–Bn modules
- Emulation of a subset of the VAX standard console (automatic or manual bootstrap, automatic or manual restart, and a simple command language for examining or altering the state of the processor)
- Booting from supported Q22-bus devices, DSSI devices, and Ethernet
- Multilingual capability

The firmware is described in detail in Chapter 3.

1.2.8 Q22-Bus Interface

The Q22-bus interface is implemented by the CQBIC chip. The CQBIC chip contains approximately 40,870 transistors in a 132-pin CERQUAD surface mount package. It supports up to 16-word block mode transfers between a Q22-bus DMA device and main memory, and up to 2-word block mode transfers between the CPU and Q22-bus devices. It has a 420-ns cycle time for longword read transfers and an 560-ns cycle time for quadword read transfers. It has a 140-ns cycle time for unmasked longword writes and a 490-ns cycle time for masked longword writes. The Q22-bus interface contains the following:

- A 16-entry map cache for the 8,192-entry scatter/gather map that resides in main memory, used for translating 22-bit Q22-bus addresses into 26-bit main memory addresses
- Interrupt arbitration logic that recognizes Q22-bus interrupt requests BR7–BR4

The Q22-bus interface handles programmed and power-up resets, and CPU halts (deassertion of DCOK).

The KA660–AA and –BA modules each contain 240-ohm termination for the Q22-bus.

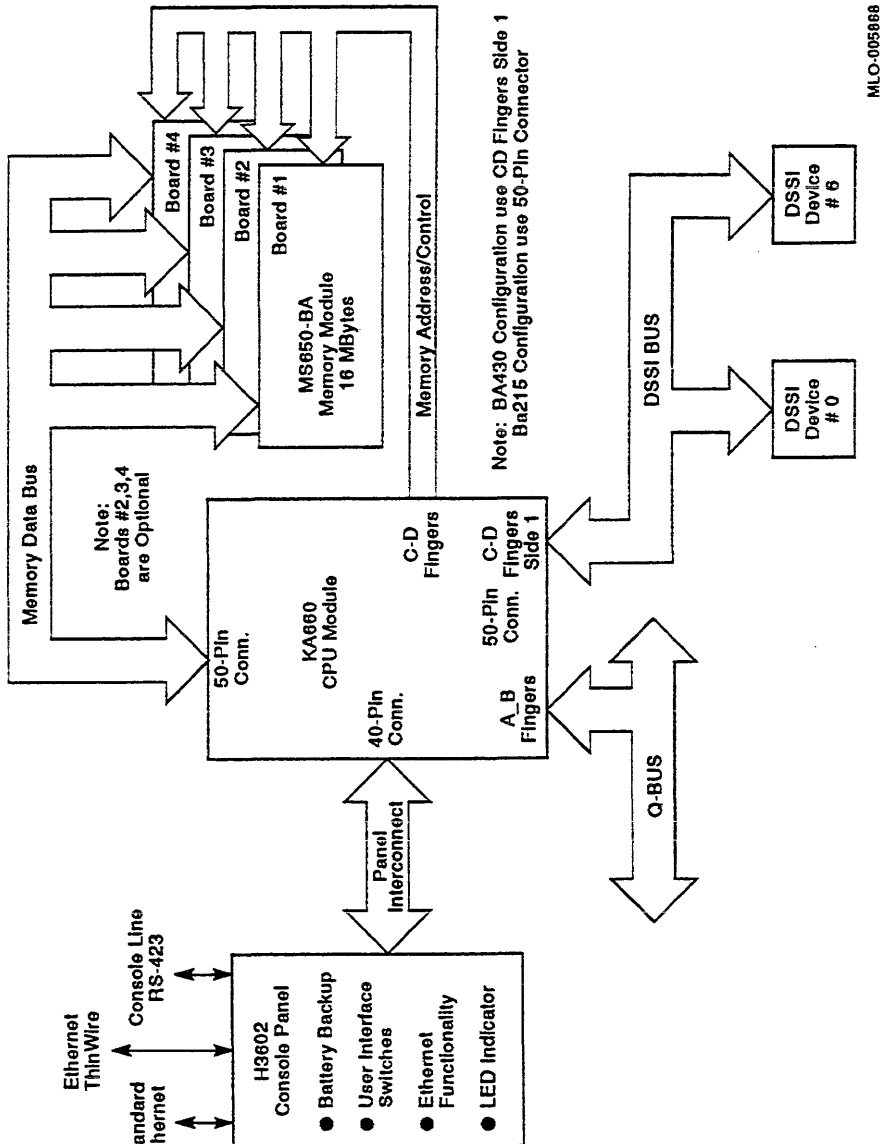
1.2.9 KA660 Ethernet Interface

The KA660 features an on-board network interface implemented through a second-generation Ethernet chip (SGEC) and a 32 x 8-bit wide ROM. This interface allows the KA660 to be connected to either a ThinWire or standard Ethernet cable through the CPU cover panel. Consult the *KA660 CPU Technical Manual* for a description.

1.2.10 KA660 DSSI Interface

See Figure 1–2 for a block diagram of the KA660 system CPU.

Figure 1–2: KA660 System CPU Block Diagram



The KA660 contains a Single-Host Adapter Chip (SHAC) chip that implements the Digital Storage Systems Interconnect (DSSI) bus interface. The DSSI interface allows the KA660 to transmit packets of data to, and receive packets of data from, up to seven other DSSI devices (RF-series disk drives or a second KA660 module). The DSSI bus improves system performance for two reasons:

- It is faster than the Q22-bus.
- It relieves the Q22-bus of disk traffic, allowing more bandwidth for Q22-bus devices.

The physical characteristics of the DSSI bus are as follows:

- 4-Mbytes-per-second bandwidth
- Distributed arbitration
- Synchronous operation
- Parity checking
- 6-meter total bus length (19.8 ft.) (includes internal and external cabling)

  Configurations that exceed this length may be supported, if properly tested
- Single-ended bus transceivers
- Maximum of eight nodes (KA660 counts as one)

  The KA660 CPU systems support four DSSI enclosures
- Eight data lines
- One parity line
- Eight control lines

Refer to the following sections for more information about the DSSI bus and disk drives:

Section 2.4	Setting and changing DSSI node names, addresses and unit numbers, dual host configuration rules
Section 3.9.14	Console SET command
Section 4.4	DSSI drive acceptance testing
Section 4.8	RF30 drive resident diagnostics and local programs

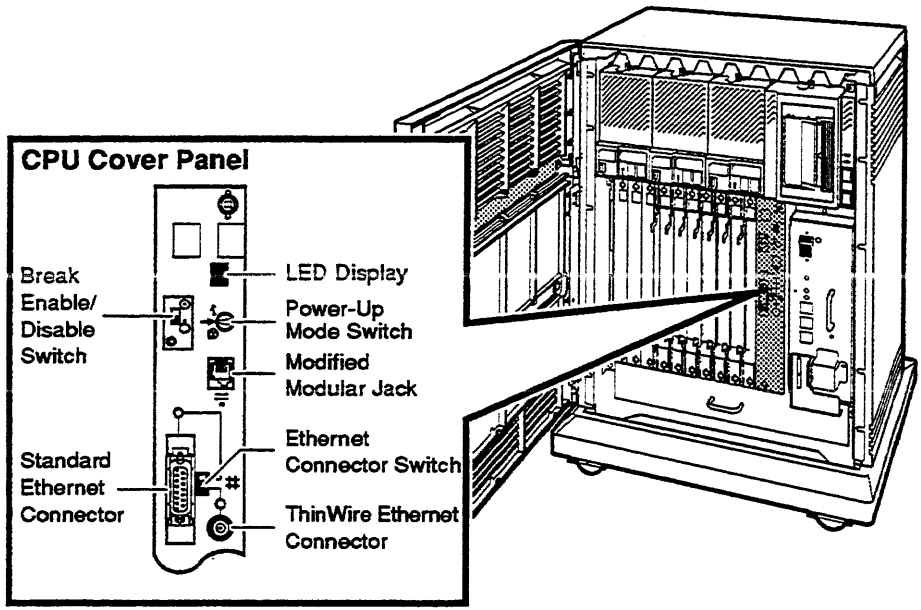
1.3 CPU Cover Panel (H3602–00)

The CPU cover panel (H3602–00, Figure 1–3) contains the console serial line connector, console baud rate switch, two Ethernet connectors and LEDs, hexadecimal LED display, and Power-Up Mode switch. The switches are read by the firmware when the processor halts. For this reason, changing the baud rate on the cover panel does not take effect until the next power-up or system reset. The switches are also read when the Power-Up Mode switch is in the test position. The cover panel has the following switches, connectors, and indicators:

- Baud rate select switch, on the back side of the panel.
- Power-Up Mode switch.
- Break Enable/Disable switch from the console keyboard BREAK key or CTRL/P, depending on the state of SSCCR <15>. Break Enable is the default.

  If this switch is set to the enable position, the system does not autoboot on power-up. It enters console I/O mode and displays the >>> prompt.
- Ethernet Connectors. The CPU cover panel has two connectors for Ethernet cable: a 15-conductor connector for standard Ethernet cable, and a BNC connector for a ThinWire Ethernet coaxial cable. The cover panel contains a switch to select the Ethernet connector, and LEDs to indicate the selected connector and valid +12 Vdc for the selected connector.
- Hexadecimal LED display, which provides a countdown of the system power-up self-tests. See Table 4–5 for the meaning of this display.

Figure 1–3: CPU Cover Panel (H3602–00)



1.4 MS650–Bn Memory Modules

The MS650–BA and MS650–BB memory modules are quad-height, Q22-bus modules. Timing of the MS650–BA (16 MBytes) and MS650–BB (8 Mbytes) modules is dependent upon the KA660 clock speed and CMCTL.

The MS650–AA memory module may not be used with the KA660 system CPU.

The KA660 and the MS650–Bn memory modules are connected through the CD rows of backplane slots 1 through 5, and through a 50-conductor cable. The part number of this cable varies depending on the number of connectors, as follows:

Number of Connectors	CPU/Memory Configuration	Part Number
3	KA660 + 2 MS650–Bn modules	17–01898–01
4	KA660 + 3 MS650–Bn modules	17–01898–02[1]
5	KA660 + 4 MS650–Bn modules	17–01898–03

[1]Recommended cable. Use five-connector cable only if this cable is not available.

The cable is keyed so that it is installed in the correct connector on the KA660 (the connector next to the module). The DSSI cable is attached to the connector "piggy backed" to the memory connector.

1.5 RF-Series ISE

The RF30 and RF31 ISEs are half-height, 13.3-cm (5.25-in) ISEs, for BA215, BA213, or BA430 enclosures. The RF71 and RF72 ISEs are full-height ISEs for BA430 enclosures.

The RF-series ISE is based on the Digital Storage Systems Interconnect (DSSI) architecture. DSSI supports up to seven storage devices, daisy-chained to the host system through the KA660 CPU or a host adapter module.

The disk drive controller is built into the RF-series ISE, rather than being a separate module. This feature enables many drive functions to be handled without host-system or adapter intervention, resulting in improved I/O performance and throughput rates.

DSSI node ID switches are located on the electronics controller module. These switches give each ISE on the DSSI bus a unique node ID number.

The RF-series ISE contains three indicators: Ready, Write-protect, and Fault.

The Ready indicator displays the activity status of the drive. It lights on power-up. After successful completion of the power-up diagnostics, the indicator goes out, until the media heads are on the requested cylinder and the drive is read/write ready.

When lit, the Write-protect indicator means the ISE is write-protected.

The Fault indicator lights at power-up. After successful completion of the power-up diagnostics, this indicator goes out. If the Fault indicator lights again after going out, a read/write safety error or a drive error condition has occurred.

Chapter 2
Configuration

2.1 Introduction

This chapter describes the guidelines for changing the configuration of a KA660 system, and for configuring a multihost system.

Before you change the system configuration, you must consider the following factors:

Module order in the backplane
Module configuration
Mass storage device configuration

If you are adding a device to a system, you must know the capacity of the system enclosure in the following areas:

Backplane
I/O panel
Power supply
Mass storage devices

2.2 General Module Order

The order of modules in the backplane depends on four factors:

- Relative use of devices in the system
- Expected performance of each device relative to other devices
- The ability of a device to tolerate delays between bus requests and bus grants (called delay tolerance or interrupt latency)
- The tendency of a device to prevent other devices farther from the CPU from accessing the bus

2.2.1 Module Order for KA660 Systems

Observe the following rules about module order:

- Install the KA660 CPU in slot 1.
- Install MS650 memory modules in slots 2, 3, 4, and 5.
- Install all Q22-bus modules in the AB rows; single-height grant cards in the A row only. Do not install dual-height modules in the CD rows, which do not route the Q22-bus.

Here is the recommended module order in a KA660 system:

KA660–AA, –BA
MS650–BA, –BB
AAV11-SA
ADV11-SA
AXV11-SA
KWV11-SA
DRV1J-SA
KMV1A-SA/SB/SC
DMV11-SA
LNV21-SF
DEQNA/DELQA/DESQA-SA
DPV11-SA
DIV32-SA
VCB02-J/H/K
DZQ11-SA
DFA01-AB
CXM04-M
CXA16-AA
CXY08-AA
CXB16-AA
CXF32-AA/AB
LPV11-SA
DRV1W-SA
KRQ50-SA
IEQ11-SA
ADQ32-SA
DRQ3B-SA
DSV11-SY
KLESI-SA
IBQ01-SA
TSV05-S
KDA50-SE
KFQSA-SE
KZQSA-SA

TQK50-SA
TQK70-SA
RQDX3-SA
KDA50-SA/KFQSA-SA
M9060-YA

2.3 Module Configuration

Each module in a system must use a unique device address and interrupt vector. The device address is also known as the control and status register (CSR) address. Most modules have switches or jumpers for setting the CSR address and interrupt vector values. The value of a floating address depends on what other modules are housed in the system.

Set CSR addresses and interrupt vectors for a module by determining the correct values for the module with the CONFIGURE command at the console I/O prompt (>>>).

The CONFIG utility eliminates the need to boot the VMS operating system to determine CSRs and interrupt vectors. Enter the CONFIGURE command, then HELP for the list of supported devices:

```
>>>CONFIGURE
Enter device configuration, HELP, or EXIT
Device,Number? help
Devices:
 LPV11      KXJ11      DLV11J     DZQ11      DZV11      DFA01
 RLV12      TSV05      RXV21      DRV11W     DRV11B     DPV11
 DMV11      DELQA      DEQNA      DESQA      RQDX3      KDA50
 RRD50      RQC25      KFQSA-DISK TQK50      TQK70      TU81E
 RV20       KFQSA-TAPE KMV11      IEQ11      DHQ11      DHV11
 CXA16      CXB16      CXY08      VCB01      QVSS       LNV11
 LNV21      QPSS       DSV11      ADV11C     AAV11C     AXV11C
 KWV11C     ADV11D     AAV11D     VCB02      QDSS       DRV11J
 DRQ3B      VSV21      IBQ01      IDV11A     IDV11B     IDV11C
 IDV11D     IAV11A     IAV11B     MIRA       ADQ32      DTC04
 DESNA      IGQ11      DIV32      KIV32      DTCN5      DTC05
 KWV32      KZQSA
```

See the description of the CONFIGURE command in Chapter 3 (Section 3.9.2) for an example of obtaining the correct CSR addresses and interrupt vectors using this command.

The LPV11–SA, which is the LPV11 version compatible with the BA200-series and BA400-series enclosures, has two sets of CSR address and interrupt vectors. To determine the correct values for an LPV11–SA, enter LPV11,2 at the DEVICE prompt for one LPV11–SA, or enter LPV11,4 for two LPV11–SA modules.

2.4 DSSI Configuration

Each device must have a unique DSSI node ID. The ISE receives its node ID from a plug on the system control panel (SCP). By convention, DSSI drives are mounted from right to left.

For more information on ISE node names, unit numbers, and other parameters, as well as information on the DUP server utility, see Appendix B.

If the cable between the ISE and the SCP is disconnected, the ISE reads the node ID from three DIP switches on its electronics controller module (ECM).

NOTE: *Pressing the system Reset button on the power supply has no effect on the ISEs. You must turn off the system and turn it back on.*

The node ID switches are located behind the 50-pin connector on the ECM. Switch 1 (the MSB) is nearest to the connector. Switch 3 (the LSB) is farthest from the connector. Table 2–1 lists the switch settings for the eight possible node addresses.

Table 2–1: ISE DIP Switch Settings

Node ID	S1	S2	S3
0	Down	Down	Down
1	Down	Down	Up
2	Down	Up	Down
3	Down	Up	Up
4	Up	Down	Down
5	Up	Down	Up
6	Up	Up	Down
7	Up	Up	Up

The VMS operating system creates DSSI disk device names according to the following scheme:

`nodename $ DIA unit number.` For example, `SUSAN$DIA3`

You can use the device name for booting, as follows:

```
>>> BOOT SUSAN$DIA3
```

You can access local programs in the ISE through the MicroVAX Diagnostic Monitor (MDM), or through the VMS operating system (version 5.4.1 or later) and console I/O mode SET HOST/DUP command. This command creates a virtual terminal connection to the storage device and the

designated local program using the Diagnostic and Utilities Protocol (DUP) standard dialog. See Appendix B for the procedure for accessing DUP through the VMS operating system. Section 3.9.14 describes the console I/O mode SET HOST/DUP command.

The KA660 DSSI node address is configured by three jumpers (W1, W2, and W3) that are found on the KA660 module as illustrated in Figure 1–1. Table 2–2 lists the jumper positions and node IDs.

Table 2–2: Setting the KA660 Node ID

Node ID	W3	W2	W1
0	Out	Out	Out
1	Out	Out	In
2	Out	In	Out
3	Out	In	In
4	In	Out	Out
5	In	Out	In
6	In	In	Out
7	In	In	In

2.4.1 DSSI Cabling for the BA215 Enclosure

The BA430 enclosure has no internal DSSI cabling. The connections are all realized by means of the backplane.

For the BA215, the cabling runs as follows:

A 50-conductor ribbon cable connects the ISE drive to the DSSI bus. A separate 5-conductor cable carries +5 Vdc and +12 Vdc to the drive from the enclosure power supply.

A 2-conductor cable connects the fifth pin on the ISE power connector to the SCP.

These cables carry the ACOK signal (same as POK) to the ISE. The SCP delays this signal to one ISE for each power supply to stagger the startup of one of two possible devices attached to each supply. This delay prevents excessive current draw at power-up. The BA215 enclosure has only one power supply, but implements this signal delay in the same way.

The 50-conductor DSSI ribbon cable connects to a 50-conductor round cable that is routed through the bottom of the mass storage area to the DSSI connector on the KA660.

CAUTION: *When removing or installing new drives, be sure to connect the rightmost connector of the DSSI ribbon cable to the round cable connected*

to the KA660. Do not "T" the bus by connecting the round connector to any of the ribbon cable's center connectors.

2.4.1.1 DSSI Bus Termination and Length

The DSSI bus must be terminated at both ends. The KA660 module terminates the DSSI bus at one end. The DSSI bus terminates at a 50-conductor connector on the left side of the enclosure. The terminator at this external connector can be removed to expand the bus.

The DSSI bus has a maximum length of 6 m (19.8 ft), including internal and external cabling.

In a dual-host system, the second KA660 module provides the bus termination.

2.4.2 Dual-Host Capability

An ISE has a multihost capability built into the firmware, which allows the drive to maintain connections with more than one DSSI adapter. Since the KA660 CPU has a built-in DSSI adapter, more than one KA660 CPU can be connected to the same DSSI bus, allowing each KA660 to access all other drives on the bus.

The primary application for such a configuration is a VAXcluster system using Ethernet as the interconnect medium between the boot and the satellite members. This configuration improves system availability, as described below.

Two KA660 systems are connected through an external DSSI cable (BC21M). Each KA660 system is a boot member for a number of satellite nodes. The system disk resides in the first enclosure, and serves as the system disk for both KA660 systems. The KA660 in each enclosure has equal access to the system disk, and to any other DSSI disk in either enclosure.

If one of the KA660 modules fails, all satellite nodes booted through that KA660 module lose connections to the system disk. However, the multihost capability enables each satellite node to know that the system disk is still available through a different path—that of the remaining good KA660 module. A connection through that KA660 is then established, and the satellite nodes are able to continue operation.

Thus, even if one KA660 module fails, the satellites booted through it are able to continue operation. The entire cluster will run in a degraded condition, since one KA660 is now serving the satellite nodes of both KA660s. Processing can continue, however, until Customer Services can repair the problem.

A dual-host system cannot recover from the following conditions:

- System disk failure. If there is only one system disk, its failure causes the entire cluster to stop functioning until the disk failure is corrected. Disk failure can be caused by such factors as a power supply failure in the enclosure containing the disk.
- DSSI cabling failure. If a failure in one of the DSSI cables renders access to the disks impossible, the cable must be repaired in order to continue operation. Since the DSSI bus cabling is not redundant, a cable failure usually results in a system failure.

2.4.3 Dual-Host Configuration

Dual-host systems have the following configuration limitations:

- A maximum of two systems can be connected, because of cabling and enclosure limitations.
- The DSSI bus supports eight devices or adapters. Since a dual-host system has two KA660 modules, and each has a connection to the DSSI bus, a maximum of six DSSI devices can be attached to the bus. See the *VAX 4000 Dual-Host Systems* manual, (EK–390AB–DH–002) for a complete list of supported dual-host configurations.
- Set DSSI node IDs as follows:
  - The first (or only) KA660 is 7.
  - The second KA660 in a dual-host system is 6. Table 2–2 explains how to set the KA660 node ID.
  - The remaining devices in a dual-host system are 0–5.

2.5 Configuration Worksheet

Use the worksheet in Figure 2–1 or Figure 2–2 to make sure the configuration does not exceed the system's limits for expansion space, I/O space, and power.

Table 2–3 lists power values for supported devices. To check a system configuration, follow these steps:

1. List all the devices to be installed in the system.
2. Fill in the information from Table 2–3 for each device.
3. Add up the columns. Make sure the totals are within the limits for the enclosure.

Table 2–3: KA660 Power and Bus Loads

		Current (Amps) (Max)		Power (Max)	BusLoads	
Option	**Module**	**+5 V**	**+12 V**	**Watts**	**AC[1]**	**DC**
AAV11–SA	A1009–PA	2.10	0.00	10.50	2.5	0.5
ADQ32–SA	A030	4.45	0.00	22.25	2.5	0.5
ADV11–SA	A1008–PA	2.00	0.00	10.00	2.3	0.5
AXV11–SA	A026–PA	2.00	0.00	10.00	1.2	0.3
CXA16–M	M3118–YA	1.60	0.20	10.40	3.0	0.5
CXB16–M	M3118–YB	2.00	0.00	10.00	3.0	0.5
CXY08–M	M3119–YA	1.64	0.395	12.94	3.0	0.5
DESQA–SA	M3127–PA	2.40	0.22	14.64	2.2	0.5
DFA01–AA	M3121–PA	1.97	0.04	10.30	3.0	1.0
DIV32–M	M7528	5.5	0.0	27.5	3.9	1.0
DPV11–SA	M8020–PA	1.20	0.30	9.60	1.0	1.0
DRQ3B–SA	M7658–PA	4.50	0.00	22.50	2.0	0.5
DRV1J–SA	M8049–PA	1.80	0.00	9.00	2.0	1.0
DRV1W–SA	M7651–PA	1.80	0.00	9.00	2.0	1.0
DSV11	M3108	5.43	0.69	35.43	3.9	1.0
DTQNA–BC	M7130	6.00	2.00	54.00	3.9	0.5
IBQ01–SA	M3125–PA	5.00	0.30	28.60	4.6	1.0
IEQ11–SA	M8634–PA	3.50	0.00	17.50	2.0	1.0
KA660–A/B[2]	M7626–A/B	6.0	0.14	32.88	3.5	1.0
KDA50–SE	M7164	6.93	0.00	34.65	3.0	0.5
KDA50–	M7165	6.57	0.03	33.21	–	–
KLESI–SA	M7740–PA	3.20	0.00	15.00	2.3	1.0
KMV1A–SA	M7500–PA	2.6	0.20	15.40	3.0	1.0
KFQSA–SE	M7769	5.50	0.00	27.50	4.4	0.5
KRQ50–SA	M7552	2.70	0.00	13.50	2.7	1.0
KWV11–SA	M4002–PA	2.20	0.013	11.156	1.0	0.3
KXJ11–SF	M7616	6.00	1.40	46.80	2.7	1.0
KZQSA–SA	M5976–SA	5.4	0.0	27.0	4.75	1.4
LPV11–SA	M8086–PA	2.80	0.00	14.00	1.8	0.5
MRV11–D	M8578	1.60[2]	0.00	8.00	3.0	0.5
MS650–BA	M7621	1.1	0.0	5.5	0.0	0.0
MS650–BB	M7621	3.9	0.0	19.53	0.0	0.0
RF31E–AA	–	1.25	2.21	27.4	–	–
RF71E–AA	–	1.25	1.64	25.93	–	–
RV20	–	3.0	–	35.3.0	0.0	–

[1]AC bus load must not exceed 22 A.
[2]Value is for the unpopulated module only.

Table 2–3 (Cont.): KA660 Power and Bus Loads

		Current (Amps) (Max)		Power (Max)	BusLoads	
Option	**Module**	**+5 V**	**+12 V**	**Watts**	**AC[1]**	**DC**
TLZ04–JA	–	2.20	0.345	15.2	–	–
TK70E–AA	–	1.50	2.40	36.30	–	–
TQK70–SA	M7559	3.2	0.00	15.0	4.3	0.5
TSV05–SA	M7530	6.50	0.00	32.50	1.5	1.0
TSV05–SA	M7206–PA	6.50	0.00	32.50	2.4	1.0

[1]AC bus load must not exceed 22 A.

NOTE: *Slot 0 will always be occupied by the M9715–AA, which generates 0.1 A @ +5V dc and 1.0 A @ +12V dc, with a total power of 12.5 W.*

NOTE: *The BA215 supports only the half-height ISEs.*

Figure 2–1: VAX 4000 Model 200 (BA430) Configuration Worksheet

Slot	Module	Current (Amps) +5 Vdc	+12 Vdc	-3.3 Vdc	-12 Vdc	Power (Watts)	Bus Load AC	DC
0								
CPU 1								
Mem 2								
Mem 3								
Mem 4								
Mem 5								
Q/CD 6								
Q/CD 7								
Q/CD 8								
Q/CD 9								
Q/CD 10							—	—
Q/CD 11								
Q/CD 12								
Mass Storage:								
Tape							—	—
1								
2								
3								
Total these columns:								
Must not exceed:		60.0 A	22.0 A	15.0 A	3.0 A	584.0 W	31	20

Note: Total output power from +3.3 Vdc and +5 Vdc must not exceed 330 W.

MLO-005711

Figure 2–2: VAX 4000 Model 200 (BA215) Configuration Worksheet

Primary Power Supply

Slot	Module	Current (Amps) +5 Vdc	Current (Amps) +12 Vdc	Power (Watts)	Bus Load AC	Bus Load DC
CPU 1						
Mem 2						
Q/CD 3						
Q/CD 4						
Q/CD 5						
Q/CD 6						
Mass Storage:						
Tape Drive						
Fixed Disk 0						
Fixed Disk 1						
Total these columns:						
Must not exceed:		33.0 A	7.6 A	230.0 W		

MLO-005712

Chapter 3
KA660 Firmware

3.1 Introduction

This chapter describes the KA660 firmware, which gains control of the processor whenever the KA660 performs a processor halt. A processor halt transfers control to the firmware. The processor does not actually stop executing instructions.

3.2 KA660 Firmware Features

The firmware is located in two 128-Kbyte EPROMS on the KA660. The firmware address range is 20040000 through 2007FFFF, in the KA660 local I/O space. The firmware displays diagnostic progress and error reports on the KA660 LEDs and on the console terminal. It provides the following features:

- Automatic or manual restart or bootstrap of customer application images at power-up, reset, or conditionally after processor halts. (Restart in this context is not the same as restarting or resetting the hardware.)
- Automatic or manual bootstrap of an operating system following processor halts.
- An interactive command language that allows you to examine and alter the state of the processor.
- Diagnostics that test all components on the board and verify that the module is working correctly.
- Support of various terminals and devices as the system console.
- Multilingual support. The firmware can issue system messages in several languages.

The processor must be functioning at a level able to execute instructions from the console program ROM for the console program to operate.

The firmware consists of the following major functional areas:

Halt entry and dispatch code
Bootstrap
Console I/O mode
Diagnostics

The halt entry and dispatch code, bootstrap, and console I/O mode are described in this chapter. Diagnostics are described in Chapter 4.

3.3 Halt Entry and Dispatch Code

The processor enters the halt entry code at physical address 20040000 whenever a halt occurs. The halt entry code saves machine state, then transfers control to the firmware halt dispatcher.

After a halt, the halt entry code saves the current LED code, then writes an E to the LEDs. An E on the LEDs indicates that at least several instructions have been successfully executed, although if the CPU is functioning properly, it occurs too quickly to be seen. The halt entry code saves the following registers. The console intercepts any direct reference to these registers and redirects it to the saved copies:

R0–R15	General purpose registers
PR$_SAVPSL	Saved processor status longword register
PR$_SCBB	System control block base register
DLEDR	Diagnostic LED register
ADxMAT	SSC address match register
ADxMAT	SSC address mask register

The halt entry code unconditionally sets the following registers to fixed values on any halt, to ensure that the console itself can run and to protect the module from physical damage.

SSCCR	SSC configuration register
ADxMAT	SSC address match register
ADxMSK	SSC address mask register
CBTCR	CDAL bus timeout control register
TIVRx	SSC timer interrupt vector registers

The console command interpreter does not modify actual processor registers. Instead it saves the processor registers in console memory when it enters the halt entry code, then directs all references to the processor registers to the corresponding saved values, not to the registers themselves.

When the processor reenters program mode, the saved registers are restored and any changes become operative only then. References to processor memory are handled normally. The binary load and unload command (X, Section 3.9.19) cannot reference the console memory pages.

After saving the registers, the halt entry code transfers control to the halt dispatch code. The halt dispatch code determines the cause of the halt by reading the halt field (PR$_SAVPSL <13:08>), the processor halt action field (PR$_CPMBX <01:00>), and the Break Enable/Disable switch on the CPU cover panel. Table 3–1 lists the actions taken, in sequence. If an action fails, the next action is taken, with the exception of bootstrap, which is not attempted after diagnostic failure.

Table 3–1: Halt Action Summary

Halt Code= 3	Break/ Enable Switch	User-Defined Halt	Console Program Mailbox	Action(s)
T	1	0, 1, 3	x	Diagnostics and console commands
T	1	1	2, 4	x
T	0	x	x	Diagnostics; if successful, boot, if fails, use console commands
F	1	0	0	Console
F	0	0	0	Restart; if this fails boot; if boot fails use console commands
F	x	1	0	Restart; if this fails, use console commands
F	x	2	0	Boot; if this fails, use console commands
F	x	3	0	Console
F	x	4	0	Restart; if this fails, boot; if boot fails, use console commands
F	x	x	1	Restart; if this fails, use console commands
F	x	x	2	Boot; if this fails, use console commands
F	x	x	3	Console

T = TRUE—indicates a Reset or Power-up condition.
F = FALSE—indicates a HALT instruction or error halt condition.
x = DON'T CARE—indicates that the condition is "don't care".

3.4 External Halts

Several conditions can trigger an external halt, and different actions are taken depending on the condition. The conditions are listed below.

- The Break Enable/Disable switch is set to enable, and you press BREAK on the system console terminal.

- Assertion of the BHALT line on the Q-bus.
- Deassertion of DCOK. A halt is delivered if the processor is not running out of halt-protected space, and the BHALT ENB bit is set. The system restart switch deasserts DCOK. DCOK may also be deasserted by the DESQA sanity timer, or any other Q22-bus module that chooses to implement the Q22-bus restart/reboot protocol.

The KA660 cannot detect the deassertion of DCOK when in console I/O mode, so no action is taken.

CAUTION: *Do not press the Restart button while in console I/O mode. Doing so will destroy system state without notifying the firmware.*

The action taken by the halt dispatch code on a console BREAK or Q22-bus BHALT is the same: the firmware enters console I/O mode if halts are enabled.

The halt dispatch code distinguishes between DCOK deasserted and BHALT by assuming that BHALT must be asserted for at least 10 ms, and that DCOK is deasserted for at most 9 μs. To determine if the BHALT line is asserted, the firmware steps out into halt-unprotected space after 9 ms. If the processor halts again, the firmware concludes that the halt was caused by the BHALT and not by the deassertion of DCOK. The firmware keeps a halt-in-progress flag to tell if it is halting because of stepping out into halt-unprotected space. This flag is cleared on power-up.

3.5 Power-Up Sequence

On power-up, the firmware performs several actions. It locates and identifies the console device, performs a language inquiry, and runs the diagnostics.

Power-up actions differ, depending on the state of the Power-Up Mode switch on the CPU cover panel (Figure 1–3). The mode switch has three settings: Test, Language Inquiry, and Normal. The differences are described in Sections 3.5.0.1 through 3.5.0.3.

3.5.0.1 Mode Switch Set to Test

Use the Test position on the H3602–00 to verify that the connection between the KA660 and the console terminal is good.

- To test the console terminal, insert the H3103 loopback connector into the H3602–00 console connector, and set the switch to the Test position. You must install the loopback connector to run the test.
- To test the console cable, install the H8572 connector on the end of the console cable, and insert the H3103 into the H8572.

During the test, the firmware toggles between the active and passive states. During the active state (3 seconds), the LED is set to 6. The firmware reads the baud rate and mode switch, then transmits and receives a character sequence.

During the passive state (5 seconds), the LED is set to 3.

If at any time the firmware detects an error (parity, framing, overflow, or no characters), the display hangs at 6. If the configuration switch is moved from the test position, the firmware continues as if on a normal power-up.

3.5.0.2 Mode Switch Set to Language Inquiry

If the Power-Up Mode switch is set to Language Inquiry, or the firmware detects that the contents of NVRAM are invalid, the firmware prompts you for the language to be used for displaying the following system messages:

```
Loading system software.
Failure.
Restarting system software.
Performing normal system tests.
Tests completed.
Normal operation not possible.
Bootfile.
```

The Language Selection Menu appears under the conditions listed in Table 3–2. The position of the Break Enable/Disable switch has no effect on these conditions.

Table 3–2: Language Inquiry on Power-Up or Reset

Mode	Language Not Previously Set[1]	Language Previously Set
Language Inquiry	Prompt[2]	Prompt
Normal	Prompt	No Prompt

[1]Action if contents of NVRAM invalid same as Language Not Previously Set.
[2]Prompt = Language Selection Menu displayed.

The Language Selection Menu is shown in Example 3–1. If no response is received within 30 seconds, the firmware defaults to English.

Example 3–1: Language Selection Menu

```
 1) Dansk
 2) Deutsch (Deutschland/Österreich)
 3) Deutsch (Schweiz)
 4) English (United Kingdom)
 5) English (United States/Canada)
 6) Español
 7) Français (Canada)
 8) Français (France/Belgique)
 9) Français (Suisse)
10) Italiano
11) Nederlands
12) Norsk
13) Portugues
14) Suomi
15) Svenska
 (1..15):
```

In addition, the console may prompt you for a default boot device. See Section 3.6. After the language inquiry, the firmware continues as if on a normal power-up.

3.5.0.3 Mode Switch Set to Normal

The console displays the Language Selection Menu if the mode switch is set to Normal and the contents of NVRAM are invalid. The console uses the saved console language if the mode switch is set to Normal and the contents of NVRAM are valid.

3.6 Bootstrap

The KA660 supports bootstrap of VAX/VMS, VAXELN, and MDM diagnostics.

The firmware initializes the system to a known state before dispatching to the primary virtual memory bootstrap (VMB), as follows:

1. Checks CPMBX<2>(BIP), bootstrap in progress. If it is set, bootstrap fails and the console displays the message `Failure.` in the selected console language.
2. If this is an automatic bootstrap, prints the message `Loading system software.` on the console terminal.
3. Validates the boot device name. If none exists, supplies a list of available devices and issues a boot device prompt. If you do not specify a device within 30 seconds, uses EZA0.
4. Writes a form of this boot request, including active boot flags and boot device (BOOT/R5:0 EZA0, for example), to the console terminal.
5. Sets CPMBX<2>(BIP).
6. Initializes the Q22-bus scatter/gather map.
7. Validates the PFN bitmap. If invalid, rebuilds it.
8. Searches for a 128-Kbyte contiguous block of good memory as defined by the PFN bitmap. If 128 Kbytes cannot be found, the bootstrap fails.
9. Initializes the general purpose registers:

R0	Address of descriptor of the boot device name or 0 if none specified
R2	Length of PFN bitmap in bytes
R3	Address of PFN bitmap
R4	Time-of-day of bootstrap from PR$_TODR
R5	Boot flags
R10	Halt PC value
R11	Halt PSL value (without halt code and mapenable)
AP	Halt code
SP	Base of 128-Kbyte good memory block + 512
PC	Base of 128-Kbyte good memory block + 512
R1, R6, R7, R8, R9, FP	0

10. Copies the VMB image from EPROM to local memory, beginning at the base of the 128 Kbytes of good memory block + 512.
11. Exits from the firmware to VMB residing in memory.

VMB is the primary bootstrap for VAX processors. The KA660 VMB resides in the firmware, and is copied into main memory before control is transferred to it. VMB then loads the secondary bootstrap image and transfers control to it.

3.7 Operating System Restart

An operating system restart is the process of bringing up the operating system from a known initialization state following a processor halt.

A restart occurs under the conditions listed in Table 3–1, earlier in this chapter.

To restart a halted operating system, the firmware searches system memory for the restart parameter block (RPB), a data structure constructed for this purpose by VMB. If the firmware finds a valid RPB, it passes control to the operating system at an address specified in the RPB.

The firmware keeps a restart-in-progress (RIP) flag in CPMBX which it uses to avoid repeated attempts to restart a failing operating system. The operating system maintains an additional RIP flag in the RPB.

The firmware restarts the operating system in the following sequence:

1. Checks CPMBX<3>(RIP). If it is set, restart fails.
2. Prints the message `Restarting system software.` on the console terminal.
3. Sets CPMBX<3>(RIP).
4. Searches for a valid RPB. If none is found, restart fails.
5. Checks the operating system RPB$L_RSTRTFLG<0>(RIP) flag. If it is set, restart fails.
6. Writes a 0 (zero) to the diagnostic LEDs.
7. Dispatches to the restart address, RPB$L_RESTART, with:

   SP = the physical address of the RPB plus 512
   AP = the halt code
   PSL = 041F0000
   PR$_MAPEN = 0.

If the restart is successful, the operating system must clear CPMBX<3>(RIP).

If restart fails, the firmware prints `Failure.` on the console terminal.

3.7.1 Locating the RPB

The RPB is a page-aligned control block that can be identified by its signature in the first three longwords:

+00 (first longword) = physical address of the RPB
+04 (second longword) = physical address of the restart routine
+08 (third longword) = checksum of first 31 longwords of restart routine

The firmware finds a valid RPB as follows:

1. Searches for a page of memory that contains its address in the first longword. If none is found, the search for a valid RPB has failed.
2. Reads the second longword in the page (the physical address of the restart routine). If it is not a valid physical address, or if it is zero, returns to step 1. The check for zero is necessary to ensure that a page of zeros does not pass the test for a valid RPB.
3. Calculates the 32-bit two's-complement sum (ignoring overflows) of the first 31 longwords of the restart routine. If the sum does not match the third longword of the RPB, returns to step 1.
4. If the sum matches, a valid RPB has been found.

3.8 Console I/O Mode

In console I/O mode several characters have special meaning, as listed in Table 3–3.

3.8.1 Command Syntax

The console accepts commands up to 80 characters long. Longer commands produce error messages. The character count does not include rubouts, rubbed-out characters, or the RETURN at the end of the command.

You can abbreviate a command by entering only as many characters as are required to make the command unique. Most commands can be recognized from their first character.

The console treats two or more consecutive spaces and tabs as a single space. Leading and trailing spaces and tabs are ignored. You can place command qualifiers after the command keyword or after any symbol or number in the command.

All numbers (addresses, data, counts) are hexadecimal, but symbolic register names contain decimal register numbers. The hexadecimal digits are 0 through 9 and A through F. You can use uppercase and lowercase letters in hexadecimal numbers (A through F) and commands.

Table 3–3: Console I/O Mode Special Characters

RETURN	(Carriage Return) This character ends a command line. No action is taken on a command until after it is terminated by a carriage return. A null line terminated by a carriage return is treated as a valid, null command. No action is taken, and the console re-prompts for input. Carriage return is echoed as carriage return, line feed.
X	(Delete Character) When the operator types rubout, the console deletes the character that the operator previously typed. What appears on the console terminal depends on whether the terminal is a video terminal or a hardcopy terminal. For hard copy terminals, when a rubout is typed, the console echoes with a backslash ("<backslash>"), followed by the character being deleted. If the operator types additional rubouts, the additional characters deleted are echoed. When the operator types a non-rubout character, the console echoes another backslash, followed by the character typed. The result is to echo the characters deleted, surrounding them with backslashes. For video terminals, when RUBOUT is typed the previous character is erased from the screen and the cursor is restored to its previous position. The console does not delete characters past the beginning of a command line. If the operator types more rubouts than there are characters on the line, the extra rubouts are ignored. If a RUBOUT is typed on a blank line, it is ignored.
Ctrl/A	(or F14) Toggle insertion/overstrike mode for command line editing. By default, the console powers up to overstrike mode.
Ctrl/B	(or up_arrow or down_arrow) Recall previous command(s). Command recall is only operable if sufficient memory is available. This function may then be enabled and disabled using the SET RECALL command.
Ctrl/C	Control-C causes the console to echo ^C and to abort processing of a command. Control-C has no effect as part of a binary load data stream. Control-C clears control-S, and reenables output stopped by control-O.
Ctrl/D	(Control-D or left_arrow) This character moves the cursor left one position.
Ctrl/E	(Control-E) Move cursor to the end of the line.
Ctrl/F	(Control-F or right_arrow) Moves the cursor right one position.
Ctrl/H	(Control-H, BACKSPACE or F12) Moves cursor to the beginning of the line.
Ctrl/O	(Control-O) This character causes the console to throw away transmissions to the console terminal until the next control-O is entered. Control-O is echoed as ^O<CR> when it disables output, but is not echoed when it reenables output. Output is reenabled if the console prints an error message, or if it prompts for a command from the terminal. Displaying a REPEAT command does not reenable output. When output is reenabled for reading a command, the console prompt is displayed. Output is also enabled control-S.
Ctrl/Q	(Control-Q) This character causes the output to the console terminal to resume. Additional control-Q's are ignored. Control-S and control-Q are not echoed.

Ctrl/S	(Control-S) Stops output to the console terminal until control-Q is typed. Control-S and control-Q are not echoed.
Ctrl/U	(Control-U) The console echoes ^U<CR>, and deletes the entire line. If control-U is typed on an empty line, it is echoed, and the console prompts for another command.
Ctrl/R	(Control-R) Causes the console to echo <CR><LF> followed by the current command line. This function can be used to improve the readability of a command line that has been heavily edited. When control-C is typed as part of a command line, the console deletes the line as it does with control-U.
Ctrl/P	(Control-P) If in console I/O mode, causes the console to echo ^P and to abort processing of a command. If the console is in program I/O mode control-P is passed to the operating system.
BREAK	(BREAK) If the console is in console I/O mode, BREAK is equivalent to control-C and is echoed as "^C".

NOTE: *If the local console is in program I/O mode and halts are disabled, BREAK is ignored. If the console is in program I/O mode and halts are enabled, BREAK causes the processor to halt and enter console I/O mode.*

3.8.2 Address Specifiers

Several commands take an address or addresses as arguments. An address defines the address space, and the offset into that space. The console supports six address spaces:

Physical memory
Virtual memory
Protected memory
General purpose registers (GPR)
Internal processor registers (IPR)
The PSL

The address space that the console references is inherited from the previous console reference, unless you explicitly specify another address space. The initial address space is physical memory.

3.8.3 Symbolic Addresses

The console supports symbolic references to addresses. A symbolic reference defines the address space, and the offset into that space. Table 3–4 lists symbolic references supported by the console, grouped according to address space. You do not have to use an address space qualifier when using a symbolic address.

Table 3–4: Console Symbolic Addresses

Symbol	Address	Symbol	Address
GPR Address Space (/G)			
R0	0	R1	1
R2	2	R3	3
R4	4	R5	5
R6	6	R7	7
R8	8	R9	9
R10	0A	R11	0B
R12	0C	R13	0D
R14	0E	R15	0F
AP	0C	FP	0D
SP	0D	PC	0E
PSL	–	–	–
IPR Address Space (/I)			
pr$_ksp	00	pr$_esp	01
pr$_ssp	02	pr$_usp	03
pr$_isp	04	pr$_p0br	08
pr$_p0lr	09	pr$_p1br	0A
pr$_p1lr	0B	pr$_sbr	0C
pr$_slr	0D	pr$_pcbb	10
pr$_scbb	11	pr$_ipl	12
pr$_astlv	13	pr$_sirr	14
pr$_sisr	15	pr$_iccr	18
pr$_nicr	19	pr$_icr	1A
pr$_todr	1B	pr$_rxcs	20
pr$_rxdb	21	pr$_txcs	22
pr$_txdb	23	pr$_tbdr	24
pr$_ccr	25	pr$_mcesr	26
pr$_mser	27	pr$_savpc	2A
pr$_savpsl	2B	pr$_ioreset	37
pr$_mapen	38	pr$_tbia	39
pr$_tbis	3A	pr$_sid	3E
pr$_tbchk	3F	–	–

Table 3–4 (Cont.): Console Symbolic Addresses

Symbol	Address	Symbol	Address
Physical Memory (/P)			
qbio	20000000	qbmem	30000000
qbmbr	20080010	—	—
rom	20040000	cacr	20084000
bdr	20084004	—	—
dscr	20080000	dser	20080004
dmear	20080008	dsear	2008000C
ipcr0	20001f40	ipcr1	20001f42
ipcr2	20001f44	ipcr3	20001f46
ssc_ram	20140400	ssc_cr	20140010
ssc_cdal	20140020	ssc_dledr	20140030
ssc_ad0mat	20140130	ssc_ad0msk	20140134
ssc_ad1mat	20140140	ssc_ad1msk	20140144
ssc_tcr0	20140100	ssc_tir0	20140104
ssc_tnir0	20140108	ssc_tivr0	2014010c
ssc_tcr1	20140110	ssc_tir1	20140114
ssc_tnir1	20140118	ssc_tivr1	2014011c
memcsr0	20080100	memcsr1	20080104
memcsr2	20080108	memcsr3	2008010c
memcsr4	20080110	memcsr5	20080114
memcsr6	20080118	memcsr7	2008011c
memcsr8	20080120	memcsr9	20080124
memcsr10	20080128	memcsr11	2008012c
memcsr12	20080130	memcsr13	20080134
memcsr14	20080138	memcsr15	2008013c
memcsr16	20080140	memcsr17	20080144
nicsr0	20008000	nicsr1	20008004
—	20008008	nicsr3	2000800C
nicsr4	20008010	nicsr5	20008014
nicsr6	20008018	nicsr7	2000801C
—	20008020	nicsr9	20008024
nicsr10	20008028	nicsr11	2000802C
nicsr12	20008030	nicsr13	20008034
nicsr14	20008038	nicsr15	2000803C
sgec_setup	20008000	sgec_poll	20008004
—	20008008	sgec_rba	2000800C
sgec_tba	20008010	sgec_status	20008014
sgec_mode	20008018	sgec_sbr	2000801C

Table 3–4 (Cont.): Console Symbolic Addresses

Symbol	Address	Symbol	Address
Physical Memory (/P)			
—	20008020	sgec_wdt	20008024
sgec_mfc	20008028	sgec_verlo	2000802C
sgec_verhi	20008030	sgec_proc	20008034
sgec_bpt	20008038	sgec_cmd	2000803C
shac_sswcr	20004230	shac_sshma	20004244
shac_pqbbr	20004048	shac_psr	2000404c
shac_pesr	20004250	shac_pfar	20004254
shac_ppr	20004058	shac_pmcsr	2000405C
shac_pcq0cr	20004280	shac_pcq1cr	20004284
shac_pcq2cr	20004088	shac_pcq3cr	2000408C
shac_pdfqcr	20004290	shac_pmfqcr	20004294
shac_psrcr	20004098	shac_pecr	2000409C
shac_pdcr	200042A0	shac_picr	200042A4
shac_pmtcr	200040A8	shac_pmtecr	200040AC

Table 3–5 lists symbolic addresses that can be used in any address space.

Table 3–5: Symbolic Addresses Used in Any Address Space

Symbol	Description
*	The location last referenced in an EXAMINE or DEPOSIT command.
+	The location immediately following the last location referenced in an EXAMINE or DEPOSIT command. For references to physical or virtual memory spaces, the location referenced is the last address, plus the size of the last reference (1 for byte, 2 for word, 4 for longword, 8 for quadword). For other address spaces, the address is the last address referenced plus one.
—	The location immediately preceding the last location referenced in an EXAMINE or DEPOSIT command. For references to physical or virtual memory spaces, the location referenced is the last address minus the size of this reference (1 for byte, 2 for word, 4 for longword, 8 for quadword). For other address spaces, the address is the last address referenced minus one.
@	The location addressed by the last location referenced in an EXAMINE or DEPOSIT command.

3.8.4 Console Command Qualifiers

You can enter console command qualifiers in any order on the command line after the command keyword. The three types of qualifiers are: data control, address space control, and command specific. Table 3–6 lists and describes the data control and address space control qualifiers. Command specific qualifiers are described in Section 3.9.

Table 3–6: Console Command Qualifiers

Qualifier	Description
Data Control	
/B	The data size is byte.
/W	The data size is word.
/L	The data size is longword.
/Q	The data size is quadword.
/N:{count}	An unsigned hexadecimal integer that is evaluated into a longword. This qualifier determines the number of additional operations that are to take place on EXAMINE, DEPOSIT, MOVE, and SEARCH commands. An error message appears if the number overflows 32 bits.
/STEP:{size}	Step. Overrides the default increment of the console current reference. Commands that manipulate memory, such as EXAMINE, DEPOSIT, MOVE, and SEARCH, normally increment the console current reference by the size of the data being used.
/WRONG	Wrong. Used to override or set error bits when referencing main memory. On writes, use the complement. On reads, ignore ECC errors.
Address Space Control	
/G	General purpose register (GPR) address space, R0–R15. The data size is always longword.
/I	Internal processor register (IPR) address space. Accessible only by the MTPR and MFPR instructions. The data size is always longword.
/V	Virtual memory address space. All access and protection checking occur. If access to a program running with the current PSL is not allowed, the console issues an error message. Deposits to virtual space cause the PTE<M> bit to be set. If memory mapping is not enabled, virtual addresses are equal to physical addresses. Note that when you examine virtual memory, the address space and address in the response is the physical address of the virtual address.
/P	Physical memory address space.
/M	Processor status longword (PSL) address space. The data size is always longword.
/U	Access to console private memory is allowed. This qualifier also disables virtual address protection checks. On virtual address writes, the PTE<M> bit is not set if the /U qualifier is present. This qualifier is not inherited; it must be respecified on each command.

3.8.5 Console Command Keywords

Table 3–7 lists command keywords by type. Table 3–8 lists the parameters, qualifiers, and arguments for each console command. Parameters, used with the SET and SHOW commands only, are listed in the first column along with the command.

Although it is possible to abbreviate by using the minimum number of characters required to uniquely identify a command or parameter, these abbreviations may become ambiguous at a later time if a new command or parameter is added in an updated version of the firmware. For this reason you should not use abbreviations in programs.

Table 3–7: Command Keywords by Type

Processor Control	Data Transfer	Console Control
B*OOT	D*EPOSIT	CONF*IGURE
C*ONTINUE	E*XAMINE	F*IND
H*ALT	M*OVE	R*EPEAT
I*NITIALIZE	SEA*RCH	SET
N*EXT	X	SH*OW
S*TART		T*EST
U*NJAM		

Qualifier Keywords

Data Control	Address Space Control	Command Specific
/B	/G	/IN*STRUCTION
/W	/I	/NO*T
/L	/P	/R5: or /
/Q	/V	/RP*B or /ME*M
/N:	/M	/F*ULL
/ST*EP:	/U	/DU*P or /MA*INTENANCE
/WR*ONG		/DS*SI or /U*QSSP
		/DI*SK or /T*APE
		/SE*RVICE

"*" indicates the minimal number of characters that are required to uniquely identify the keyword.

Table 3–8: Console Command Summary

Command	Qualifiers	Argument	Other(s)
BOOT	/R5:{boot_flags} /{boot_flags}	[{boot_device}]	—
CONFIGURE	—	—	—
CONTINUE	—	—	—
DEPOSIT	/B /W /L /Q — /G /I /V /P /M /U /N:{count} /STEP:{size} /WRONG :{ecc}]	{address}	{data} [{data}]
EXAMINE	/B /W /L /Q — /G /I /V /P /M /U /N:{count} /STEP:{size} /WRONG /INSTRUCTION	[{address}]	—
FIND	/MEM /RPB	—	—
HALT	—	—	—
HELP	—	—	—
INITIALIZE	—	—	—
MOVE	/B /W /L /Q — /V /P /U /N:{count} /STEP:{size} /WRONG [:{ecc}]	{src_address}	{dest_address}
NEXT	—	[{count}]	—
REPEAT	—	{command}	—
SEARCH	/B /W /L /Q — /V /P /U /N:{count} /STEP:{size} /WRONG /NOT	{start_address}	{pattern} [{mask}]
SET BFL(A)G	—	{bitmap}	—
SET BOOT	—	{device_string}	—
SET CONTROLP	—	{0/1}	—
SET HALT	—	{halt_action}	—
SET HOST	/DUP /DSSI /BUS:{0/1}	{node_number}	[{task}]
SET HOST	/DUP /UQSSP {/DISK ! /TAPE } /DUP /UQSSP	{controller_number} {csr_address}	[{task}] [{task}]
SET HOST	/MAINTENANCE /UQSSP /SERVICE /MAINTENANCE /UQSSP	{controller_number} {csr_address}	
SET LANGUAGE	—	{language_type}	—
SET RECALL	—	{0/1}	—
SET VERIFICATION	—	{password}	—
SHOW BFL(A)G	—	—	—
SHOW BOOT	—	—	—
SHOW CONTROLP	—	—	—
SHOW DEVICE	—	—	—
SHOW DSSI	—	—	—

Table 3–8 (Cont.): Console Command Summary

Command	Qualifiers	Argument	Other(s)
SHOW ETHERNET	—	—	—
SHOW HALT	—	—	—
SHOW LANGUAGE	—	—	—
SHOW MEMORY	/FULL	—	—
SHOW QBUS	—	—	—
SHOW RECALL	—	—	—
SHOW RLV12	—	—	—
SHOW SCSI	—	—	—
SHOW TRANSLATION	—	{phys_address}	—
SHOW UQSSP	—	—	—
SHOW VERIFICATION	—	—	—
SHOW VERSION	—	—	—
START	—	{address}	—
TEST	—	{test_number}	[{parameters}]
UNJAM	—	—	—
X	—	{address}	{count}

3.9 Console Commands

This section describes the console I/O mode commands. Enter the commands at the console I/O mode prompt >>>.

3.9.1 BOOT

The BOOT command initializes the processor and transfers execution to VMB. VMB attempts to boot the operating system from the specified device, or the default boot device if none is specified. The console qualifies the bootstrap operation by passing a boot flags bitmap to VMB in R5. Table 3–9 lists the supported R5 boot flags.

Format:

BOOT [qualifier-list] [boot_device]

If you do not enter either the qualifier or the device name, then the default value is used. Explicitly stating the boot flags or the boot device overrides but does not permanently change the corresponding default value.

Set the default boot device and boot flags with the SET BOOT and SET BFLAG commands. If you do not set a default boot device, the processor times out after 30 seconds and attempts to boot from the on-board Ethernet port, EZA0. Table 3–10 lists the boot devices supported by the KA660–AA.

Command Specific Qualifiers:

/R5:{bitmap}	A 32-bit hexadecimal value passed to VMB in R5. The console does not interpret this value. Use the SET BFLAG command to specify a default boot flags longword. Use the SHOW BFLAG command to display the longword.
/{bitmap}	Same as /R5:{bitmap}
[boot_device or device list]	A character string of up to 39 characters. Longer strings cause a VAL TOO BIG error message. Apart from length, the console makes no attempt to interpret or validate the device name. The console converts the string to uppercase, then passes VMB a string descriptor to this device name in R0.

Table 3–9: VMB Boot Flags

Bit	Name	Description
0	RPB$V_CONV	Conversational boot. At various points in the system boot procedure, the bootstrap code solicits parameters and other input from the console terminal.
2	RPB$V_INIBPT	Initial breakpoint. If RPB$V_DEBUG is set, the VMS operating system executes a BPT instruction in module INIT immediately after enabling mapping.
3	RPB$V_BBLOCK	Secondary bootstrap from bootblock. When set, VMB reads logical block number 0 of the boot device and tests it for conformance with the bootblock format. If in conformance, the block is executed to continue the bootstrap. No attempt is made to perform a Files–11 bootstrap.
4	RPB$V_DIAG	Diagnostic bootstrap. When set, the load image requested over the network is [SYS0.SYSMAINT]DIAGBOOT.EXE.
5	RPB$V_BOOBPT	Bootstrap breakpoint. When set, a breakpoint instruction is executed in VMB and control is transferred to XDELTA before booting.
6	RPB$V_HEADER	Image header. When set, VMB transfers control to the address specified by the file's image header. When not set, VMB transfers control to the first location of the load image.
8	RPB$V_SOLICT	File name solicit. When set, VMB prompts the operator for the name of the application image file. The maximum file specification size is 17 characters.
9	RPB$V_HALT	Halt before transfer. When set, VMB halts before transferring control to the application image.
31:28	RPB$V_TOPSYS	This field can be any value from 0 through F. This flag changes the top-level directory name for system disks with multiple operating systems. For example, if TOPSYS is 1, the top-level directory name is [SYS1...].

3.9.1.1 Supported Boot Devices

Table 3–10 lists the boot devices supported by the KA660 CPU. The table correlates the boot device names expected in a BOOT command with the corresponding supported devices.

Boot device names consist of a device code at least two letters (A through Z) in length, followed by a single character controller letter (A through Z), and ending in a device unit number (0–16,383).

DSSI device names may also include a node prefix, consisting of either a node number (0–7) or a node name (a string of up to eight characters), ending in a dollar sign ($).

3.9.1.2 Boot Devices

The KA660 firmware passes the address of a descriptor of the boot device name to VMB through R0. This device name used for the bootstrap operation is one of the following:

- The local Ethernet device, if no default boot device has been specified or
- The default boot device specified at initial power-up or via a SET BOOT command, or
- The boot device name explicitly specified in a BOOT command line.

The device name may be any arbitrary character string, with a maximum length of 17 characters. Longer strings cause an error message to be issued to the console. Otherwise the console makes no attempt at interpreting or validating the device name. The console converts the string to all upper case, and passes VMB the address of a string descriptor for the device name in R0.

Table 3–10 correlates the boot device names expected in a BOOT command with the corresponding supported devices.

Table 3–10: Boot Device Names

Device Type	Controller/Adapter	Device Logical Name
RF-series ISE	Embedded DSSI host adapter (part of CPU)	DImn[1]
RF-series ISE	KFQSA storage adapter	DUcn[2]
TK-series tape drive	TQK70/TQK50	MUcn[3]
TLZ04 tape drive	KZQSA adapter	MKAn
RRD40 compact disc drive	KZQSA adapter	DKAn
PROM (programmable read-only memory)	MRV11 module	PRAn
Ethernet adapter	On-board (part of CPU)	EZA0
Ethernet adapter	DESQA Ethernet controller	XQAn
RA-series drives	KDA50	DUcn[2]

[1] m = DSSI bus adapter (A = first bus (0), B = second bus (1), and so on.)
n = unit number
When under operating system control, DIBn devices are recognized as DIAn devices.

[2] c = MSCP controller designator (A = first, B = second, and so on.)
n = unit number

[3] c = TMSCP controller designator (A = first, B = second, and so on.)
n = unit number

Examples:

```
>>> SHOW BOOT
0
>>> SHOW BFLAG
EZA0
>>> B! Boot using default boot flags and device.
(BOOT/R5:0 EZA0)

  2..
-EZA0

>>> B XQA0 ! Boot from XQA0 using default boot flags.
(BOOT/R5:0 XQA0)

  2..
-XQA0

>>> B/10 ! Boot using supplied boot flag (4)
(BOOT/R5:10 EZA0)       ! and default device.
```

```
  2..
-EZA0

>>> BOOT /R5:220 XQA0    ! Boot using supplied boot flags
(BOOT/R5:220 XQA0)       ! (5 and 9) and device.

  2..
-XQA0
```

3.9.2 CONFIGURE

The CONFIGURE command invokes an interactive mode that permits you to enter Q22-bus device names, then generates a table of Q22-bus I/O page device CSR addresses and interrupt vectors. CONFIGURE is similar to the VMS SYSGEN CONFIG utility. This command simplifies field configuration by providing information that is typically available only with a running operating system. Refer to the example below and use the CONFIGURE command as follows:

1. Enter CONFIGURE at the console I/O prompt.
2. Enter HELP at the `Device, Number?` prompt to see a list of devices whose CSR addresses and interrupt vectors can be determined.
3. Enter the device names and number of devices.
4. Enter EXIT to obtain the CSR address and interrupt vector assignments.

The devices listed in the HELP display are not necessarily supported by the KA660–AA CPU.

Format:

CONFIGURE

Example:

```
>>> CONFIGURE
Enter device configuration, HELP, or EXIT
Device,Number? HELP
Devices:
 LPV11          KXJ11          DLV11J         DZQ11          DZV11          DFA01
 RLV12          TSV05          RXV21          DRV11W         DRV11B         DPV11
 DMV11          DELQA          DEQNA          DESQA          RQDX3          KDA50
 RRD50          RQC25          KFQSA-DISK     TQK50          TQK70          TU81E
 RV20           KFQSA-TAPE     KMV11          IEQ11          DHQ11          DHV11
 CXA16          CXB16          CXY08          VCB01          QVSS           LNV11
 LNV21          QPSS           DSV11          ADV11C         AAV11C         AXV11C
 KWV11C         ADV11D         AAV11D         VCB02          QDSS           DRV11J
 DRQ3B          VSV21          IBQ01          IDV11A         IDV11B         IDV11C
 IDV11D         IAV11A         IAV11B         MIRA           ADQ32          DTC04
 DESNA          IGQ11          DIV32          KIV32          DTCN5          DTC05
 KWV32          KZQSA
Numbers:
```

```
 1 to 255, default is 1
Device,Number? kda50
Device,Number? kfqsa
Device is ambiguous
Device,Number? kfqsa-disk
Device,Number? kfqsa-tape
Device,Number? cxy08
Device,Number? cxa16
Device,Number? exit

Address/Vector Assignments
-772150/154 KDA50
-760334/300 KFQSA-DISK
-774500/260 KFQSA-TAPE
-760500/310 CXY08
-760520/320 CXA16

>>>
```

3.9.3 CONTINUE

The CONTINUE command causes the processor to begin instruction execution at the address currently contained in the PC. It does not perform a processor initialization. The console enters program I/O mode.

Format:

CONTINUE

Example:

```
>>> CONTINUE
```

3.9.4 DEPOSIT

The DEPOSIT command deposits data into the address specified. If you do not specify an address space or data size qualifier, the console uses the last address space and data size used in a DEPOSIT, EXAMINE, MOVE, or SEARCH command. After processor initialization, the default address space is physical memory, the default data size is longword, and the default address is zero. If you specify conflicting address space or data sizes, the console ignores the command and issues an error message.

Format:

DEPOSIT [qualifier_list] {address} {data} [data...]

Qualifiers:

Data control: /B, /W, /L, /Q, /N:{count}, /STEP:{size}, /WRONG

Address space control: /G, /I, /P, /V, /U

Arguments:

{address} A longword address that specifies the first location into which data is deposited. The address can be an actual address or a symbolic address.

{data} The data to be deposited. If the specified data is larger than the deposit data size, the firmware ignores the command and issues an error response. If the specified data is smaller than the deposit data size, it is extended on the left with zeros.

[data] Additional data to be deposited (as many as can fit on the command line).

Examples:

```
>>> D/P/B/N:1FF 0 0     ! Clear first 512 bytes of physical memory.
>>> D/V/L/N:3 1234 5    ! Deposit 5 into four longwords starting
                        ! at virtual memory address 1234.
>>> D/N:8 R0 FFFFFFFF   ! Loads GPRs R0 through R8 with -1.
```

```
>>> D/N:200 - 0          ! Starting at previous address, clear 513
                         ! bytes.
>>> D/L/P/N:10/S:200 0 8  ! Deposit 8 in the first longword of
                          ! the first 17 pages in physical
                          ! memory.
```

3.9.5 EXAMINE

The EXAMINE command examines the contents of the memory location or register specified by the address. If no address is specified, + is assumed. The display line consists of a single character address specifier, the physical address to be examined, and the examined data.

EXAMINE uses the same qualifiers as DEPOSIT. However, the /WRONG qualifier causes examines to ignore ECC errors on reads from physical memory. The EXAMINE command also supports an /INSTRUCTION qualifier, which will disassemble the instructions at the current address.

Format:

EXAMINE [qualifier_list] [address]

Qualifiers:

Data control: /B, /W, /L, /Q, /N:{count}, /STEP:{size}, /WRONG, /INSTRUCTION

Address space control: /G, /I, /M, /P, /V, /U

Command specific:

/INSTRUCTION	Disassembles and displays the VAX Macro-32 instruction at the specified address.

Arguments:

[address]	A longword address that specifies the first location to be examined. The address can be an actual or a symbolic address. If no address is specified, + is assumed.

Examples:

```
>>> EX PC                          ! Examine the PC.
  G 0000000F FFFFFFFC
>>> EX SP                          ! Examine the SP.
  G 0000000E 00000200
>>> EX PSL                         ! Examine the PSL.
  M 00000000 041F0000
>>> E/M                            ! Examine PSL another way.
  M 00000000 041F0000
>>> E R4/N:5                       ! Examine R4 through R9.
  G 00000004 00000000
  G 00000005 00000000
  G 00000006 00000000
  G 00000007 00000000
  G 00000008 00000000
  G 00000009 801D9000

>>> EXPR$_SCBB                   ! Examine the SCBB, IPR 17
  I 00000011 2004A000            ! (decimal).

>>> E/P 0                          ! Examine local memory 0.
  P 00000000 00000000

>>> EX /INS 20040000              ! Examine 1st byte of ROM.
  P 20040000   11 BRB     20040019

>>> EX /INS/N:5 20040019          ! Disassemble from branch.
  P 20040019   D0 MOVL    I^#20140000,@#20140000
  P 20040024   D2 MCOML   @#20140030,@#20140502
  P 2004002F   D2 MCOML   S^#0E,@#20140030
  P 20040036   7D MOVQ    R0,@#201404B2
  P 2004003D   D0 MOVL    I^#201404B2,R1
  P 20040044   DB MFPR    S^#2A,B^44(R1)

>>> E /INS                              ! Look at next instruction.
  P 20040048   DB MFPR    S^#2B,B^48(R1)
>>>
```

3.9.6 FIND

The FIND command searches main memory starting at address zero for a page-aligned 128-Kbyte segment of good memory, or a restart parameter block (RPB). If the command finds the segment or RPB, its address plus 512 is left in SP (R14). If it does not find the segment or RPB, the console issues an error message and preserves the contents of SP. If you do not specify a qualifier, /RPB is assumed.

Format:

FIND [qualifier-list]

Qualifiers:

Command specific:

/MEMORY Searches memory for a page-aligned block of good memory, 128 Kbytes in length. The search looks only at memory that is deemed usable by the bitmap. This command leaves the contents of memory unchanged.

/RPB Searches all of physical memory for an RPB. The search does not use the bitmap to qualify which pages are looked at. The command leaves the contents of memory unchanged.

Examples:

```
>>> EX SP                          ! Check the SP.
  G 0000000E 00000000
>>> FIND /MEM                      ! Look for a valid 128 Kbyte.
>>> EX SP                          ! Note where it was found.
  G 0000000E 00000200
>>> FIND /RPB                      ! Check for valid RPB.
?2C FND ERR 00C00004               ! None to be found here.
>>>
```

3.9.7 HALT

The HALT command has no effect. It is included for compatibility with other VAX consoles.

Format:

HALT

Example:

```
>>> HALT                  ! Pretend to halt.
>>>
```

3.9.8 HELP

The HELP command provides information about command syntax and usage. *Example:*

```
>>>HELP

Following is a brief summary of all commands supported by the console:

        UPPERCASE  denotes a keyword that you must type in
        |          denotes an OR condition
        []         denotes optional parameters
        <>         denotes a field specifying a syntactically correct value
        ..         denotes one of an inclusive range of integers
        ...        denotes that the previous item may be repeated
```

```
Valid qualifiers:
    /B /W /L /Q /INSTRUCTION
    /G /I /V /P /M
    /STEP: /N: /NOT
    /WRONG /U

Valid commands:
    BOOT [/R5:<boot_flags> | /<boot_flags>]
 [<boot_device>[:]]
    CONFIGURE
    CONTINUE
    DEPOSIT [<qualifiers>] <address> [<datum>
 [<datum>]]
    EXAMINE [<qualifiers>] [<address>]
    FIND [/MEMORY | /RPB]
    HALT
    HELP
    INITIALIZE
    MOVE [<qualifiers>] <address> <address>
    NEXT [count]
    REPEAT <command>
    SEARCH [<qualifiers>] <address> <pattern> [<mask>]
    SET BFL(A)G <boot_flags>
    SET BOOT <boot_device>
    SET CONTROLP <0..1 | DISABLED | ENABLED>
    SET HALT <0..4 | DEFAULT | RESTART | REBOOT | HALT | RESTART_REBOOT>
    SET HOST/DUP/DSSI <node_number> [<task>]
    SET HOST/DUP/UQSSP </DISK | /TAPE> <controller_number> [<task>]
    SET HOST/DUP/UQSSP <physical_CSR_address> [<task>]
    SET HOST/MAINTENANCE/UQSSP/SERVICE <controller_number>
    SET HOST/MAINTENANCE/UQSSP <physical_CSR_address>
    SET LANGUAGE <1..15>
    SET RECALL <0..1 | DISABLED | ENABLED>
    SHOW BFL(A)G
    SHOW BOOT
    SHOW DEVICE
    SHOW DSSI
    SHOW ETHERNET
    SHOW HALT
    SHOW LANGUAGE
    SHOW MEMORY [/FULL]
    SHOW RECALL
    SHOW RLV12
    SHOW QBUS
    SHOW UQSSP
    SHOW SCSI
    SHOW TRANSLATION <physical_address>
    SHOW VERSION
    START <address>
    TEST [<test_code> [<parameters>]]
    UNJAM
    X <address> <count>
>>>
```

3.9.9 INITIALIZE

The INITIALIZE command performs a processor initialization.

Format:

INITIALIZE

The following registers are initialized:

Register	State at Initialization
PSL	041F0000
IPL	1F
ASTLVL	4
SISR	0
ICCS	Bits <6> and <0> clear; the rest are unpredictable
RXCS	0
TXCS	80
MAPEN	0
Cache	Enabled and flushed
Instruction buffer	Uneffected
Console previous reference	Longword, physical, address 0
TODR	Uneffected
Main memory	Uneffected
General registers	Uneffected
Halt code	Uneffected
Bootstrap-in-progress flag	Uneffected
Internal restart-in-progress flag	Uneffected

The firmware clears all error status bits and initializes the following:

CDAL bus timer
Address decode and match registers
Programmable timer interrupt vectors
SSCCR

Example:

```
>>> INIT
>>>
```

3.9.10 MOVE

The MOVE command copies the block of memory starting at the source address to a block beginning at the destination address. Typically, this command has an /N qualifier so that more than one data is transferred. The destination correctly reflects the contents of the source, regardless of the overlap between the source and the data.

The MOVE command actually performs byte, word, longword, and quadword reads and writes as needed in the process of moving the data. Moves are supported only for the physical and virtual address spaces.

Format:

MOVE [qualifier-list] {src_address} {dest_address}

Qualifiers:

Data control: /B, /W, /L, /W, /N:{count}, /STEP:{size}, /WRONG

Address space control: /V, /U, /P

Arguments:

{src_address} A longword address that specifies the first location of the source data to be copied.

{dest_address} A longword address that specifies the destination of the first byte of data. These addresses may be an actual address or a symbolic address. If no address is specified, + is assumed.

Examples:

```
>>> EX/N:4 0                ! Observe destination.
P 00000000 00000000
P 00000004 00000000
P 00000008 00000000
P 0000000C 00000000
P 00000010 00000000

>>> EX/N:4 200               ! Observe source data.
P 00000200 58DD0520
P 00000204 585E04C1
P 00000208 00FF8FBB
P 0000020C 5208A8D0
P 00000210 540CA8DE

>>> MOV/N:4 200 0            ! Move the data.
```

```
>>> EX/N:4 0                    ! Observe moved data.
P 00000000 58DD0520
P 00000004 585E04C1
P 00000008 00FF8FBB
P 0000000C 5208A8D0
P 00000010 540CA8DE
>>>
```

3.9.11 NEXT

The NEXT command executes the specified number of macro instructions. If no count is specified, 1 is assumed.

After the last macro instruction is executed, the console reenters console I/O mode.

Format:

NEXT {count}

The console implements the NEXT command using the trace trap enable and trace pending bits in the PSL, and the trace pending vector in the SCB. The following restrictions apply:

- If memory management is enabled, the NEXT command works only if the first page in SSC RAM is mapped in S0 (system) space.
- Overhead associated with the NEXT command affects execution time of an instruction.
- The NEXT command elevates the IPL to 31 for long periods of time (milliseconds) while single stepping over several commands.
- Unpredictable results occur if the macro instruction being stepped over modifies either the SCBB or the trace trap entry. This means that you cannot use the NEXT command in conjunction with other debuggers.

Arguments:

{count} A value representing the number of macro instructions to execute.

Examples:

```
>>> DEP 1000 50D650D4                          ! Create a simple program.
>>> DEP 1004 125005D1
>>> DEP 1008 00FE11F9
>>> EX /INSTRUCTION /N:5 1000                  ! List it.
  P 00001000   D4 CLRL    R0
  P 00001002   D6 INCL    R0
  P 00001004   D1 CMPL    S^#05,R0
  P 00001007   12 BNEQ    00001002
  P 00001009   11 BRB     00001009
  P 0000100B   00 HALT
>>> DEP PR$_SCBB 200                           ! Set up a user SCBB...
>>> DEP PC 1000                                ! ...and the PC.
>>>
>>> N                                          ! Single step...
  P 00001002   D6 INCL    R0                  ! SPACEBAR
  P 00001004   D1 CMPL    S^#05,R0            ! SPACEBAR
  P 00001007   12 BNEQ    00001002            ! SPACEBAR
  P 00001002   D6 INCL    R0                  ! CR
>>> N 5                                        ! ...or multiple step the program.
  P 00001004   D1 CMPL    S^#05,R0
  P 00001007   12 BNEQ    00001002
  P 00001002   D6 INCL    R0
  P 00001004   D1 CMPL    S^#05,R0
  P 00001007   12 BNEQ    00001002
>>> N 7
  P 00001002   D6 INCL    R0
  P 00001004   D1 CMPL    S^#05,R0
  P 00001007   12 BNEQ    00001002
  P 00001002   D6 INCL    R0
  P 00001004   D1 CMPL    S^#05,R0
  P 00001007   12 BNEQ    00001002
  P 00001009   11 BRB     00001009
>>> N
  P 00001009   11 BRB     00001009
>>>
```

3.9.12 REPEAT

The REPEAT command repeatedly displays and executes the specified command. Press CTRL/C to stop the command. You can specify any valid console command, except the REPEAT command.

Format:

REPEAT {command}

Arguments:

{command} A valid console command other than REPEAT.

Examples:

```
>>>REPEAT EXAMINE 0
  P 00000000 00000004
  P 00000000 00000004
  P 00000000 00000004
  P 00000000 00000004
  P 00000000 00000004
  P 00000000 00000004
  P 00000000 00000004
  P 0000^C
>>>
```

3.9.13 SEARCH

The SEARCH command finds all occurrences of a pattern and reports the addresses where the pattern was found. If the /NOT qualifier is present, the command reports all addresses in which the pattern did not match.

Format:

SEARCH [qualifier_list] {address} {pattern} [mask]

SEARCH accepts an optional mask that indicates bits to be ignored (*don't care* bits). For example, to ignore bit 0 in the comparison, specify a mask of 1. The mask, if not present, defaults to 0.

A match occurs if (pattern AND mask complement) = (data AND mask complement), where:

pattern is the target data
mask is the optional don't care bitmask (which defaults to 0)
data is the data at the current address

SEARCH reports the address under the following conditions:

/NOT Qualifier	Match Condition	Action
Absent	True	Report address
Absent	False	No report
Present	True	No report
Present	False	Report address

The address is advanced by the size of the pattern (byte, word, longword, or quadword), unless overridden by the /STEP qualifier.

Qualifiers:

Data control: /B, /W, /L, /Q, /N:{count}, /STEP:{size}, /WRONG, /NOT

Address space control: /P, /V, /U

Command specific:

/NOT Inverts the sense of the match.

Arguments:

{start_address} A longword address that specifies the first location subject to the search. This address can be an actual address or a symbolic address. If no address is specified, + is assumed.

{pattern} The target data.

[{mask}] A mask of the bits desired in the comparison.

Examples:

```
9wide\maximum)

>>>DEP /P/L/N:1000 0 0     ! Clear some memory.
>>>
>>>DEP 300 12345678\BOLD)    ! Deposit some "search" data.
>>>DEP 401 12345678\BOLD)
>>>DEP 502 87654321\BOLD)
>>>
>>>SEARCH /N:1000 /ST:1 0 12345678 ! Search for all occurances...
  P 00000300 12345678     ! ...of 12345678 on any byte...
  P 00000401 12345678     ! ...boundary.
>>>SEARCH /N:1000 0 12345678  ! Then try on longword...
  P 00000300 12345678     ! ...boundaries.
>>>SEARCH /N:1000 /NOT 0 0  ! Search for all non-zero...
  P 00000300 12345678     ! ...longwords.
  P 00000400 34567800
  P 00000404 00000012
  P 00000500 43210000
  P 00000504 00008765
>>>SEARCH /N:1000 /ST:1 0 1 FFFFFFFE ! Search for "odd" longwords...
  P 00000502 87654321     ! ...on any boundary.
  P 00000503 00876543
  P 00000504 00008765
  P 00000505 00000087
>>>SEARCH /N:1000 /B 0 12   ! Search for all occurrances...
  P 00000303 12      ! ...of the byte 12.
  P 00000404 12
>>>SEARCH /N:1000 /ST:1 /w 0 FE11  ! Search for all words which...
>>>                                 ! ...could be interpretted as...
>>>                                 ! ...a "spin" (10$: brb 10$).
>>>       ! Note, none found.
```

3.9.14 SET

The SET command sets the parameter to the value you specify.

Format:

SET {parameter} {value}

Parameters:

BFLAG	Set the default R5 boot flags. The value must be a hexadecimal number of up to 8 digits. See Table 3–9 under the BOOT command description for a list of the boot flags.
BOOT	Set the default boot device. The value must be a valid device name as specified in the BOOT command description Section 3.9.1.
CONTROLP	Set Control-P as the console halt condition instead of break. Value of 1 sets Control-P; value of 0 disables Control-P.
HALT	Set the user-defined halt action; acceptable values are 0 through 4 or the following keywords: DEFAULT, RESTART, REBOOT, HALT, and RESTART_REBOOT.

HOST — Connect to the DUP or MAINTENANCE driver on the selected node or device. Note the hierarchy of the SET HOST qualifiers below.

/DUP—Use the DUP driver to execute local programs of a device on either the DSSI bus or the Q22-bus.

/DSSI node—Attach to the DSSI node. A node is a name up to 8 characters in length or a number from 0 to 7. OB
/UQSSP—Attach to the UQSSP device specified using one of the following methods:

/DISK n—Specifies the disk controller number, where n is a number from 0 to 255. The resulting fixed address for n=0 is 20001468 and the floating rank for n>0 is 26.
/TAPE n—Specifies the tape controller number, where n is a number from 0 to 255. The resulting fixed address for n=0 is 20001940 and the floating rank for n>0 is 30.
csr_address—Specifies the Q22-bus I/O page CSR address for the device.

/MAINTENANCE—Examines and modifies KFQSA EEPROM configuration parameters. Does not accept a task value.

/UQSSP— Attach to the UQSSP device specified using one of the following methods:

/SERVICE n—Specifies the KFQSA module n where n is a value from 0 to 3. (The resulting fixed address of a KFQSA module in service mode is 20001910+4*n.)
/csr_address—Specifies the Q22-bus I/O page CSR address for the KFQSA.

LANGUAGE — Sets console language and keyboard type. If the current console terminal does not support the Digital Multinational Character Set (MCS), then this command has no effect and the console message appears in English. Values are 1 through 15. Refer to Example 3–1 for the languages you can select.

RECALL — Sets command recall state to either 1 or 0 (ENABLED or DISABLED).

Qualifiers: Listed in the parameter descriptions above.

Examples:

```
>>>
>>> SET BFLAG 220
>>>
>>> SET BOOT
>>>
>>> SET CONTROLP 0
>>>
>>> SET HALT REBOOT
>>>
>>> SET HOST/DUP/DSSI 0
Starting DUP server...
```

```
DSSI Node 0 (SUSAN)
Copyright © 1990  Digital Equipment Corporation
DRVEXR V1.0   D  5-JUL-1990 15:33:06
DRVTST V1.0   D  5-JUL-1990 15:33:06
HISTRY V1.0   D  5-JUL-1990 15:33:06
ERASE  V1.0   D  5-JUL-1990 15:33:06
PARAMS V1.0   D  5-JUL-1990 15:33:06
DIRECT V1.0   D  5-JUL-1990 15:33:06
End of directory

Task Name?  params
Copyright © 1990  Digital Equipment Corporation

PARAMS>  stat path

   ID  Path Block     Remote Node      DGS_S      DGS_R      MSGS_S     MSGS_R
   --  ------------  ---------------  ---------- ---------- ---------- ----------
    0 PB  FF811ECC    Internal Path            0          0          0          0
    6 PB  FF811FD0   KFQSA  KFX V1.0          0          0          0          0
    1 PB  FF8120D4   KAREN  RFX V101           0          0          0          0
    4 PB  FF8121D8   WILMA  RFX V101           0          0          0          0
    5 PB  FF8122DC   BETTY  RFX V101           0          0          0          0
    2 PB  FF8123E0   DSSI1  VMS V5.0           0          0      14328      14328
    3 PB  FF8124E4   3      VMB BOOT           0          0         61         61

PARAMS>  exit
Exiting...

Task Name?

Stopping DUP server...
>>>
>>> SET HOST/DUP/DSSI 0 PARAMS
Starting DUP server...

DSSI Node 0 (SUSAN)
Copyright © 1990  Digital Equipment Corporation

PARAMS>  show node

Parameter      Current           Default         Type      Radix
---------  ----------------  ----------------  --------  -----
NODENAME           SUSAN             RF30        String  Ascii   B

PARAMS> SHOW ALLCLASS

Parameter      Current           Default         Type      Radix
---------  ----------------  ----------------  --------  -----
ALLCLASS                 1                 0      Byte     Dec   B

PARAMS> EXIT
Exiting...

Stopping DUP server...
>>>
>>> SET HOST /DUP/DSSI/BUS:1  0
Starting DUP server...
```

```
DSSI Bus 1 Node 0 (SUSAN)
Copyright © 1990  Digital Equipment Corporation
DRVEXR V1.0  D  5-JUL-1990 15:33:06
DRVTST V1.0  D  5-JUL-1990 15:33:06
HISTRY V1.0  D  5-JUL-1990 15:33:06
ERASE  V1.0  D  5-JUL-1990 15:33:06
PARAMS V1.0  D  5-JUL-1990 15:33:06
DIRECT V1.0  D  5-JUL-1990 15:33:06
End of directory

Task Name? params
Copyright © 1990  Digital Equipment Corporation

PARAMS>  stat path

   ID  Path Block     Remote Node      DGS_S      DGS_R      MSGS_S     MSGS_R
   --- ------------  ---------------  ---------- ---------- ---------- ----------
    0 PB  FF811ECC   Internal Path            0          0          0          0
    6 PB  FF811FD0  KFQSA  KFX V1.0           0          0          0          0
    1 PB  FF8120D4  KAREN  RFX V101           0          0          0          0
    4 PB  FF8121D8  WILMA  RFX V101           0          0          0          0
    5 PB  FF8122DC  BETTY  RFX V101           0          0          0          0
    2 PB  FF8123E0  DSSI1  VMS V5.0           0          0      14328      14328
    3 PB  FF8124E4  3      VMB BOOT           0          0         61         61

PARAMS>  exit
Exiting...

Task Name?

Stopping DUP server...
>>>
>>> SET HOST /DUP/DSSI/BUS:1 0 PARAMS
Starting DUP server...

DSSI Bus 1 Node 0 (SUSAN)
Copyright © 1990  Digital Equipment Corporation

PARAMS> show node

Parameter     Current          Default        Type     Radix
--------- ---------------  ---------------  --------  -----
NODENAME          SUSAN             RF31      String  Ascii   B

PARAMS>  show allclass

Parameter     Current          Default        Type     Radix
--------- ---------------  ---------------  --------  -----
ALLCLASS                1                0      Byte    Dec   B

PARAMS> exit
Exiting...

Stopping DUP server...
>>>
>>> SET HOST /MAINT/UQSSP 20001468
UQSSP Controller (772150)
```

```
Enter SET, CLEAR, SHOW, HELP, EXIT, or QUIT
Node   CSR Address   Model
 0       772150       21
 1       760334       21
 4       760340       21
 5       760344       21
 7     ------ KFQSA ------
? HELP
Commands:
    SET <node> /KFQSA                        set KFQSA DSSI node number
    SET <node> <CSR_address> <model>         enable a DSSI device
    CLEAR <node>                             disable a DSSI device
    SHOW                                     show current configuration
    HELP                                     print this text
    EXIT                                     program the KFQSA
    QUIT                                     don't program the KFQSA
Parameters:
    <node>                                   0 to 7
    <CSR_address>                            760010 to 777774
    <model>                                  21 (disk) or 22 (tape)
?  set 6 /kfqsa
?  show
Node   CSR Address   Model
 0       772150       21
 1       760334       21
 4       760340       21
 5       760344       21
 6     ------ KFQSA ------
?  EXIT
Programming the KFQSA...
>>>
>>> SET LANGUAGE 5
>>>
>>> SET RECALL 1
>>>
>>> SET VERIFICATION CATNMOUSE
```

3.9.15 SHOW

The SHOW command displays the console parameter you specify.

Format:

SHOW {parameter}

Parameters:

BFLAG	Displays the default R5 boot flags.
BOOT	Displays the default boot device.
CONTROLP	Displays current state of halt recognition, either ENABLED or DISABLED.
DEVICE	Displays all devices displayed by the SHOW DSSI, SHOW ETHERNET, and SHOW UQSSP commands.

DSSI	Displays the status of all nodes that can be found on the DSSI bus. For each node on the DSSI bus, the firmware displays the node number, the node name, and the boot name and type of the device, if available. The command does not indicate if the device contains a bootable image. The node that issues the command is listed with a node name of * (asterisk). The device information is obtained from the media type field of the MSCP command GET UNIT STATUS. If a node is not running or is not capable of running an MSCP server, then no device information is displayed.
ETHERNET	Displays hardware Ethernet address for all Ethernet adapters that can be found, both on-board and on the Q22-bus. Displays as blank if no Ethernet adapter is present.
HALT	Show the user-defined halt action.
LANGUAGE	Displays console language and keyboard type. Refer to the corresponding SET LANGUAGE command for the meaning.
MEMORY	Displays main memory configuration board by board. **/FULL**—Additionally, displays the normally inaccessible areas of memory, such as the PFN bitmap pages, the console scratch memory pages, the Q22-bus scatter/gather map pages. Also reports the addresses of bad pages, as defined by the bitmap.
QBUS	Displays all Q22-bus I/O addresses that respond to an aligned word read, and vector and device name information. For each address, the console displays the address in the VAX I/O space in hexadecimal, the address as it would appear in the Q22-bus I/O space in octal, and the word data that was read in hexadecimal. This command may take several minutes to complete. Press CTRL/C to terminate the command. During execution, the command disables the scatter/gather map.
RECALL	Shows the current state of command recall, either ENABLED or DISABLED.
RLV12	Displays all RL01 and RL02 disks that appear on the Q22-bus.

SCSI — Shows any SCSI devices on the system.

TRANSLATION — Shows any virtual addresses that map to the specified physical address. The firmware uses the current values of page table base and length registers to perform its search; it is assumed that page tables have been properly built.

UQSSP — Displays the status of all disks and tapes that can be found on the Q22-bus that support the UQSSP protocol. For each such disk or tape on the Q22-bus, the firmware displays the controller number, the controller CSR address, and the boot name and type of each device connected to the controller. The command does not indicate if the device contains a bootable image.

This information is obtained from the media type field of the MSCP command GET UNIT STATUS. The console does not display device information if a node is not running (or cannot run) an MSCP server.

VERSION — Displays the current firmware version.

Qualifiers: Listed in the parameter descriptions above. *Examples:*

```
>>>
>>> SHOW BFLAG
00000220
>>>
>>> SHOW BOOT
>>>
>>> SHOW DEVICE

DSSI Bus 0 Node 0 (SUSAN)
-DIA0 (RF31)

DSSI Bus 0 Node 1 (KAREN)
-DIA1 (RF31)

DSSI Bus 0 Node 3 (*)

DSSI Bus 0 Node 4 (WILMA)
-DIA4 (RF31)

DSSI Bus 0 Node 5 (BETTY)
-DIA5 (RF31)

DSSI Bus 0 Node 6 (KFQSA)

SCSI Adapter 0 (761300), SCSI ID 7
-DKA100 (DEC RZ31      (C) DEC)
-DKA300 (MAXTOR XT-8000S)

UQSSP Disk Controller 0 (772150)
-DUA0 (RF31)

UQSSP Disk Controller 1 (760334)
-DUB1 (RF31)

UQSSP Disk Controller 2 (760340)
-DUC3 (RF31)

UQSSP Disk Controller 3 (760344)
-DUD4 (RF31)
```

```
Ethernet Adapter
-(08-00-2B-03-82-78)
>>>
>>> SHOW DSSI
DSSI Bus 0 Node 0 (SUSAN)
-DIA0 (RF31)

DSSI Bus 0 Node 1 (KAREN)
-DIA1 (RF31)

DSSI Bus 0 Node 3 (*)

DSSI Bus 0 Node 4 (WILMA)
-DIA4 (RF31)

DSSI Bus 0 Node 5 (BETTY)
-DIA5 (RF31)

DSSI Bus 0 Node 6 (KFQSA)
>>>
>>> SHOW ETHERNET
Ethernet Adapter
-(08-00-2B-03-82-78)
>>>
>>> SHOW HALT
Reboot
>>> SHOW LANGUAGE
English (United States/Canada)
>>>
>>> SHOW MEMORY
Memory 0: 00000000 to 003FFFFF, 4MB, 0 bad pages

Total of 4MB, 0 bad pages, 98 reserved pages
>>>
>>> SHOW MEMORY /FULL
Memory 0: 00000000 to 003FFFFF, 4MB, 0 bad pages

Total of 4MB, 0 bad pages, 98 reserved pages

Memory Bitmap
-003F3C00 to 003F3FFF, 2 pages

Console Scratch Area
-003F4000 to 003F7FFF, 32 pages

Qbus Map
-003F8000 to 003FFFFF, 64 pages

Scan of Bad Pages
>>>
>>> SHOW QBUS
Scan of Qbus I/O Space
-200000DC (760334) = 0000 (300) RQDX3/KDA50/RRD50/RQC25/KFQSA-DISK
-200000DE (760336) = 0AA0
-200000E0 (760340) = 0000 (304) RQDX3/KDA50/RRD50/RQC25/KFQSA-DISK
-200000E2 (760342) = 0AA0
-200000E4 (760344) = 0000 (310) RQDX3/KDA50/RRD50/RQC25/KFQSA-DISK
-200000E6 (760346) = 0AA0
-20001468 (772150) = 0000 (154) RQDX3/KDA50/RRD50/RQC25/KFQSA-DISK
-2000146A (772152) = 0AA0
-20001F40 (777500) = 0020 (004) IPCR
```

```
Scan of Qbus Memory Space
>>>
>>> SHOW RLV12
>>>
>>> SHOW SCSI

SCSI Adapter 0 (761300), SCSI ID 7
-DKA100 (DEC RZ31     (C) DEC)
-DKA300 (MAXTOR XT-8000S)
>>>
>>> SHOW TRANSLATION 1000
 V 80001000
>>>
>>> SHOW UQSSP
UQSSP Disk Controller 0 (772150)
-DUA0 (RF31)

UQSSP Disk Controller 1 (760334)
-DUB1 (RF31)

UQSSP Disk Controller 2 (760340)
-DUC4 (RF31)

UQSSP Disk Controller 3 (760344)
-DUD5 (RF31)
>>>
>>> SHOW VERIFICATION
wadenmargo
>>>
>>> SHOW VERSION
KA660-A V4.0, VMB 2.12
>>>
```

3.9.16 START

The START command starts instruction execution at the address you specify. If no address is given, the current PC is used. If memory mapping is enabled, macro instructions are executed from virtual memory, and the address is treated as a virtual address. The START command is equivalent to a DEPOSIT to PC, followed by a CONTINUE. It does not perform a processor initialization.

Format:

START [{address}]

Arguments:

[address]	The address at which to begin execution. This address is loaded into the user's PC.

Example:

```
>>> START 1000
```

3.9.17 TEST

The TEST command invokes a diagnostic test program specified by the test number. If you enter a test number of 0 (zero), all tests allowed to be executed from the console terminal are executed. The console accepts an optional list of up to five additional hexadecimal arguments.

Refer to Chapter 4 for a detailed explanation of the diagnostics.

Format:

TEST [test_number [test_arguments]]

Arguments:

{test_number}	A two-digit hexadecimal number specifying the test to be executed.
{test_arguments}	Up to five additional test arguments. These arguments are accepted but they have no meaning to the console.

Examples:

```
KA660-A T3.5-14, VMB 2.12
Performing normal system tests.
95..94..93..92..91..90..89..88..87..86..85..84..83..82..81..80..
79..78..77..76..75..74..73..72..71..70..69..68..67..66..65..64..
63..62..61..60..59..58..57..56..55..54..53..52..51..50..49..48..
47..46..45..44..43..42..41..40..39..38..37..36..35..34..33..32..
31..30..29..28..27..26..25..24..23..22..21..20..19..18..17..16..
15..14..13..12..11..10..09..08..07..06..05..04..03..
Tests completed.

>>>>>>T 9E
Test
#   Address   Name                Parameters
-------------------------------------------------------------------------------
    20052400  SCB
    20053314  De_executive
30  2005D07C  MS650_Init_Bitmap   *** mark_Hard_SBEs ******
31  2005CE3C  MS650_Setup_CSRs    **********
32  2005C984  CMCTL regs          MEMCSR0_addr *********
33  2005C940  CMCTL_powerup       *
34  20054A24  SSC_ROM             *
3F  2005EEBC  MS650_FDM_Addr_shorts *** cont_on_err ******
40  2005F6A4  MS650_count_pages   First_board Last_bd Soft_errs_allowed *******
41  20062718  Board_Reset         *
42  20054AE4  Chk_for_Interrupts  *****
43  2005AD88  SOC_DI_Cache_w_mem  cache_config start_add end_add add_incr ******
44  2005A5BC  SOC_D_Cache_w_Mem   cache_config start_add end_add add_incr ******
45  2005A21C  SOC_Cache_mem_CQBIC cache_config start_add end_add add_incr ******
46  2005B2A4  SOC_Cache1_diag_mode cache_config addr_incr ********
47  2005F434  MS650_Refresh       start_a end incr cont_on_err time_seconds *****
48  2005EAF4  MS650_Addr_shorts   start_add end_add * cont_on_err pat2 pat3 ****
49  2005E530  MS650_FDM           *** cont_on_err ******
4A  2005E288  MS650_ECC_SBEs      start_add end_add add_incr cont_on_err ******
4B  2005E048  MS650_Byte_Errors   start_add end_add add_incr cont_on_err ******
4C  2005DAEC  MS650_ECC_Logic     start_add end_add add_incr cont_on_err ******
4D  2005D95C  MS650_Address       start_add end_add add_incr cont_on_err ******
4E  2005D768  MS650_Byte          start_add end_add add_incr cont_on_err ******
```

```
4F  2005D4BC  MS650_Data          start_add end_add add_incr cont_on_err ******
51  200627E7  FPA                *******
52  20055090  SSC_Prog_timers     which_timer wait_time_us ***
53  20055360  SSC_TOY_Clock       repeat_test_250ms_ea Tolerance ***
54  20054BB9  Virtual_Mode       *********
55  20055512  Interval_Timer      *
58  20061370  SHAC_RESET          dssi_bus port_number time_secs
59  200604C4  SGEC_LPBCK_ASSIST  time_secs **
5A  2005A120  SOC_CMCTL           dont_report_memory_bad repeat_count *
5C  20060A2C  SHAC                shac_number *******
5F  2005F878  SGEC                loopback_type no_ram_tests ******
60  20059D51  SSC_Console_SLU     start_BAUD end_BAUD ******
62  20055950  console_QDSS        mark_not_present selftest_r0 selftest_r1 *****
63  20055ACC  QDSS_any            input_csr selftest_r0 selftest_r1 ******
80  20059731  CQBIC_memory_LMGH   **********
81  200555B4  Qbus_MSCP           IP_csr ******
82  20055779  Qbus_DELQA         device_num_addr ****
83  200569CA  QZA_LPBCK1          controller_number ********
84  20058070  QZA_LPBCK2          controller_number *********
85  20055C28  QZA_memory          incr test_pattern controller_number *******
86  200560E4  QZA_DMA             Controller_number main_mem_buf ********
87  200592B0  QZA_EXTLPBCK        controller_number ****
90  2005500E  CQBIC_registers     *
91  20054FA4  CQBIC_powerup       **
99  200629F5  Flush_Ena_Caches    dis_flush_cache *********
9A  200618FC  INTERACTION         pass_count disable_device ****
9B  200625D8  Init_memory_4MB     *
9C  2005BB4A  List_CPU_registers  *
9D  2005C826  Utility             Expnd_err_msg get_mode init_LEDs clr_ps_cnt
9E  20055586  List_diagnostics    *
9F  20062B32  Create_A0_Script    **********
C1  200546D0  SSC_RAM_Data        *
C2  200548A6  SSC_RAM_Data_Addr   *
C5  20059581  SSC_registers       *
C6  20054614  SSC_powerup         *********
C7  2005967C  SSC_CBTCR_timeout   ***
Scripts
#   Description
A0  User defined scripts
A1  Powerup tests, Functional Verify, continue on error, numeric countdown
A3  Functional Verify, stop on error, test # announcements
A4  Loop on A3 Functional Verify
A5  Address shorts test, run fastest way possible
A6  Memory tests, mark only multiple bit errors
A7  Memory tests
A8  Memory acceptance tests, mark single and multi-bit errors, call A7
A9  Memory tests, stop on error
B5  SOC Cache debug script
>>>! List all diagnostic tests
```

```
>>>>>>T 9C
  SBR=017B8000    SLR=00002021  SAVPC=20044827 SAVPSL=04190304    SCBB=20052400
 P0BR=80000000   P0LR=00100A80   P1BR=0A0A0A08   P1LR=000B0B0B     SID=14000006
 TODR=0010E085   ICCS=00000000                  MAPEN=00000000   BDMTR=20084000
 TCR0=00000000   TIR0=00000000  TNIR0=00000000  TIVR0=00000078   BDMKR=0000007C
 TCR1=00000001   TIR1=0052680A  TNIR1=0000000F  TIVR1=0000007C
 RXCS=00000000   RXDB=0000000D   TXCS=00000000   TXDB=00000030
  SCR=0000D000   DSER=00000000  QBEAR=0000000F   DEAR=00000000   QBMBR=017F8000
  BDR=08D0EFFF  DLEDR=0000000C  SSCCR=00D55537  CBTCR=00000004   IPCR0=0000
DSSI_0=00  (BUS_0)     PQBBR_0=03060022     PMCSR_0=00000000     SSHMA_0=0000CA20
     PSR_0=00000000     PESR_0=00000000     PFAR_0=00000000      PPR_0=00000000
NICSR0=1FFF0003   3=00004030   4=00004050   5=8039FF00   6=83E0F000  7=00000000
NICSR9=04E204E2  10=00040000  11=00000000  12=00000000  13=00000000 15=0000FFFF
NISA=08-00-2B-12-BC-AC     RDES0=00441300   1=00000000   2=05EE0000  3=000046F0
                           TDES0=00008C80   1=07000000   2=00400000  3=000040FA
MEM_FRU 1      MCSR_0=80000017      1=80400017      2=80800017       3=80C00017
MEM_FRU 2      MCSR_4=81000016      5=81400016      6=00000016       7=00000016
MEM_FRU 3      MCSR_8=00000000      9=00000000     10=00000000      11=00000000
MEM_FRU 4      MCSR12=00000000     13=00000000     14=00000000      15=00000000
MEMCSR17=00000013  MEMCSR16=00000044  CSR16_page_address=00000000
MSER=00000000  CCR=00000014

>>>T 9F
SP=201406A8
Script in ?[0=SSC, 2=RAM] :0
Script starts at 20140794
 36 bytes left
Test number (? for list) or script number :80
CQBIC_memory_LMGH>> Run from ?[0=ROM, 2=RAM, 3=fastest possible] (0):0
CQBIC_memory_LMGH>> Error severity ? [0,1,2,3] (02):0
CQBIC_memory_LMGH>> Console error report? [0=none,1=full] (01):0
CQBIC_memory_LMGH>> Stop script on error? [0=NO,1=YES] (01):0
CQBIC_memory_LMGH>> Repeat? [0=no,1=forever,>1=count] (00):0
CQBIC_memory_LMGH>> LED on entry (01):0
CQBIC_memory_LMGH>> Console Announcement on entry (80):1
 32 bytes left
Test number (? for list) or script number :1
No such diagnostic!
 32 bytes left
Test number (? for list) or script number :
>>>
>>>! Execute test script.

>>>T FE

 Bitmap=00FF3000, Length=00001000, Checksum=807F, Busmap=00FF8000
 Test_number=41, Subtest=00, Loop_Subtest=00, Error_type=00
 Error_vector=0000, Last_exception_PC=00000000, Severity=02
 Total_error_count=0000, Led_display=0C, Console_display=03, save_mchk_code=80
 parameter_1=00000000  2=00000000  3=00000000  4=00000000  5=00000000
 parameter_6=00000000  7=00000000  8=00000000  9=00000000 10=00000000
 previous_error=00000000, 00000000, 00000000, 00000000
 Flags=00FFFC10440E, SET_mask=FF
 Return_stack=201406D4, Subtest_pc=20062730, Timeout=00030D40
>>>
```

3.9.18 UNJAM

The UNJAM command performs an I/O bus reset, by writing a 1 (one) to IPR 55 (decimal).

Format:

UNJAM

Examples:

```
>>> UNJAM
>>>
```

3.9.19 X—Binary Load and Unload

The X command is for use by automatic systems communicating with the console.

The X command loads or unloads (that is, writes to memory, or reads from memory) the specified number of data bytes through the console serial line (regardless of console type) starting at the specified address.

Format:

X {address} {count} CR {line_checksum} {data} {data_checksum}

If bit 31 of the count is clear, data is received by the console and deposited into memory. If bit 31 is set, data is read from memory and sent by the console. The remaining bits in the count are a positive number indicating the number of bytes to load or unload.

The console accepts the command upon receiving the carriage return. The next byte the console receives is the command checksum, which is not echoed. The command checksum is verified by adding all command characters, including the checksum and separating space (but not including the terminating carriage return, rubouts, or characters deleted by rubout), into an 8-bit register initially set to zero. If no errors occur, the result is zero. If the command checksum is correct, the console responds with the console I/O prompt and either sends data to the requester or prepares to receive data. If the command checksum is in error, the console responds with an error message. The intent is to prevent inadvertent operator entry into a mode where the console is accepting characters from the keyboard as data, with no escape mechanism possible.

If the command is a load (bit 31 of the count is clear), the console responds with the console I/O prompt (>>>), then accepts the specified number of bytes of data for depositing to memory, and an additional byte of received data checksum. The data is verified by adding all data characters and the checksum character into an 8-bit register initially set to zero. If the final

content of the register is nonzero, the data or checksum is in error, and the console responds with an error message.

If the command is a binary unload (bit 31 of the count is set), the console responds with the console I/O prompt (>>>), followed by the specified number of bytes of binary data. As each byte is sent, it is added to a checksum register initially set to zero. At the end of the transmission, the two's complement of the low byte of the register is sent.

If the data checksum is incorrect on a load, or if memory or line errors occur during the transmission of data, the entire transmission is completed, then the console issues an error message. If an error occurs during loading, the contents of the memory being loaded are unpredictable.

The console represses echo while it is receiving the data string and checksums.

The console terminates all flow control when it receives the carriage return at the end of the command line in order to avoid treating flow control characters from the terminal as valid command line checksums.

You can control the console serial line during a binary unload using control characters (CTRL/C, CTRL/S, CTRL/O, and so on). You cannot control the console serial line during a binary load, since all received characters are valid binary data.

The console has the following timing requirements:

- It must receive data being loaded with a binary load command at a rate of at least one byte every 60 seconds.
- It must receive the command checksum that precedes the data within 60 seconds of the carriage return that terminates the command line.
- It must receive the data checksum within 60 seconds of the last data byte.

If any of these timing requirements are not met, then the console aborts the transmission by issuing an error message and returning to the console prompt.

The entire command, including the checksum, can be sent to the console as a single burst of characters at the specified character rate of the console serial line. The console is able to receive at least 4 Kbytes of data in a single X command.

3.9.20 !—Comment

The comment character (an exclamation point) is used to document command sequences. It can appear anywhere on the command line. All characters following the comment character are ignored.

Format: !

Examples

```
>>> ! The console ignores this line.
>>>
```

Chapter 4
Troubleshooting and Diagnostics

4.1 Introduction

This chapter contains a description of KA660 ROM-based diagnostics, acceptance test procedures, and power-up self-tests for common options.

4.2 General Procedures

Before troubleshooting any system problem, check the site maintenance guide for the system's service history. Ask the system manager two questions:

- Has the system been used before, and did it work correctly?
- Have changes been made to the system recently?

Three common problems occur when you make a change to the system:

- Incorrect cabling
- Module configuration errors (incorrect CSR addresses and interrupt vectors)
- Incorrect grant continuity

Most communications modules use floating CSR addresses and interrupt vectors. If you remove a module from the system, you may have to change the addresses and vectors of other modules.

If you change the system configuration, run the CONFIGURE utility at the console I/O prompt (>>>) to determine the CSR addresses and interrupt vectors recommended by Digital. These recommended values simplify the use of the MDM diagnostic package, and are compatible with VMS device drivers. Nonstandard addresses can be selected, but they require a special setup for use with VMS drivers and MDM.

When troubleshooting, note the status of cables and connectors before you perform each step. Label cables before you disconnect them to save time and prevent you from introducing new problems.

If the system fails (or appears to fail) to boot the operating system, check the console terminal screen for an error message. If the terminal displays an error message, see Section 4.3. Check the LEDs on the device you suspect is faulty. If no errors are indicated by the device LEDs, run the ROM-based diagnostics described in this chapter. In addition, check the following connections:

- If no message appears, make sure the console terminal and the system are on. Check the power switch on both the console terminal and the system. If the terminal has a green DC OK indicator, be sure it is on.
- Check the cabling to the console terminal.
- If you cannot get a display of any kind on the console terminal, try another terminal.
- If the system green DC OK LED remains off, check the power supply and power supply cabling.
- Check the hexadecimal display on the CPU cover panel. If the display is off, check the CPU module LEDs and the CPU cabling. If a hexadecimal error message appears on the cover panel or the module, see Section 4.3.

If the system boots successfully, but a device seems to fail or an intermittent failure occurs, check the error log first for a device problem. The failing device is usually in one of the following areas:

CPU
Memory
Mass storage
Communications devices

4.3 KA660 ROM-Based Diagnostics

The KA660 ROM-based diagnostic facility, rather than the MicroVAX Diagnostic Monitor (MDM), is the primary diagnostic tool for troubleshooting and testing of the CPU, memory, Ethernet, and DSSI subsystems. ROM-based diagnostics have significant advantages:

- Load time is virtually nonexistent.
- The boot path is more reliable.
- Diagnosis is done in a more primitive state. (MDM requires successful loading of the operating system.)

The ROM-based diagnostics can indicate several different FRUs, not just the CPU module. For example, they can isolate one of up to four memory modules as FRUs.

The diagnostics run automatically on power-up. While the diagnostics are running, the LEDs on the H3602–00 display a hexadecimal countdown of the tests from F to 3 (though not in precise reverse order) before booting the operating system, and 2 to 0 while booting the operating system. A different countdown appears on the console terminal.

The ROM-based diagnostics are a collection of individual tests with parameters that you can specify. A data structure called a *script* points to the tests. (See Section 4.3.2.) There are several field and manufacturing scripts. Qualified Customer Services personnel can also create their own scripts interactively.

A program called the *diagnostic executive* determines which of the available scripts to invoke. The script sequence varies if the KA660 is in a manufacturing environment. The diagnostic executive interprets the script to determine what tests to run, the correct order to run the tests, and the correct parameters to use for each test.

The diagnostic executive also controls tests so that errors can be detected and reported. It also ensures that when the tests are run, the machine is left in a consistent and well-defined state.

4.3.1 Diagnostic Tests

Table 4–1 shows a list of the ROM-based tests and utilities. To get this listing, enter T 9E at the console prompt (T is the abbreviation of TEST). The column headings have the following meanings:

- Test is the test code or utility code.
- Address is the test or utility's base address in ROM. This address varies. The addresses shown are examples only. If a test fails, entering T FE displays diagnostic state to the console. You can subtract the base address of the failing test from the last_exception_pc to find the index into the failing test's diagnostic listing (available on microfiche).
- Name is a brief description of the test or utility.
- Parameters shows the parameters for each diagnostic test or utility. Tests accept up to ten parameters. The asterisks (*) represent parameters that are used by the tests but that you cannot specify individually. These parameters are encoded in ROM and are provided by the diagnostic executive.

Table 4–1: Test and Utility Numbers

Test	Address[1]	Name	Parameters
	20052400	SCB	
	20053314	De_executive	
30	2005D07C	MS650_Init_Bitmap	*** mark_Hard_SBEs ******
31	2005CE3C	MS650_Setup_CSRs	**********
32	2005C984	CMCTL regs	MEMCSR0_addr *********
33	2005C940	CMCTL_powerup	*
34	20054A24	SSC_ROM	*
3F	2005EEBC	MS650_FDM_Addr_shorts	*** cont_on_err ******
40	2005F6A4	MS650_count_pages	First_board Last_bd Soft_errs_allowed *******
41	20062718	Board_Reset	*
42	20054AE4	Chk_for_Interrupts	*****
43	2005AD88	SOC_DI_Cache_w_mem	cache_config start_add end_add add_ incr ******
44	2005A5BC	SOC_D_Cache_w_Mem	cache_config start_add end_add add_ incr******
45	2005A21C	SOC_Cache_mem_CQBIC	cache_config start_add end_add add_ incr ******
46	2005B2A4	SOC_Cache1_diag_mode	cache_config addr_incr ********
47	2005F434	MS650_Refresh	start_a end incr cont_on_err time_ seconds *****
48	2005EAF4	MS650_Addr_shorts	start_add end_add * cont_on_err pat2 pat3 ****
49	2005E530	MS650_FDM	*** cont_on_err ******
4A	2005E288	MS650_ECC_SBEs	start_add end_add add_incr cont_on_ err ******
4B	2005E048	MS650_Byte_Errors	start_add end_add add_incr cont_on_ err ******
4C	2005DAEC	MS650_ECC_Logic	start_add end_add add_incr cont_on_ err ******
4D	2005D95C	MS650_Address	start_add end_add add_incr cont_on_ err ******
4E	2005D768	MS650_Byte	start_add end_add add_incr cont_on_ err ******
4F	2005D4BC	MS650_Data	start_add end_add add_incr cont_on_ err ******
51	200627E7	FPA	*******
52	20055090	SSC_Prog_timers	which_timer wait_time_us ***
53	20055360	SSC_TOY_Clock	repeat_test_250ms_ea Tolerance ***
54	20054BB9	Virtual_Mode	*********
55	20055512	Interval_Timer	*

[1]These addresses may change with different versions of the software.

Table 4–1 (Cont.): Test and Utility Numbers

Test	Address[1]	Name	Parameters
58	20061370	SHAC_RESET	dssi_bus port_number time_secs
59	200604C4	SGEC_LPBCK_ASSIST	time_secs **
5A	2005A120	SOC_CMCTL	dont_report_memory_bad repeat_count *
5C	20060A2C	SHAC	shac_number *******
5F	2005F878	SGEC	loopback_type no_ram_tests ******
60	20059D51	SSC_Console_SLU	start_BAUD end_BAUD ******
62	20055950	console_QDSS	mark_not_present selftest_r0 selftest_r1 *****
63	20055ACC	QDSS_any	input_csr selftest_r0 selftest_r1 ******
80	20059731	CQBIC_memory_LMGH	**********
81	200555B4	Qbus_MSCP	IP_csr ******
82	20055779	Qbus_DELQA	device_num_addr ****
83	200569CA	QZA_LPBCK1	controller_number ********
84	20058070	QZA_LPBCK2	controller_number *********
85	20055C28	QZA_memory	incr test_pattern controller_number *******
86	200560E4	QZA_DMA	Controller_number main_mem_buf********
87	200592B0	QZA_EXTLPBCK	controller_number ****
90	2005500E	CQBIC_registers	*
91	20054FA4	CQBIC_powerup	**
99	200629F5	Flush_Ena_Caches	dis_flush_cache *********
9A	200618FC	INTERACTION	pass_count disable_device ****
9B	200625D8	Init_memory_4MB	*
9C	2005BB4A	List_CPU_registers	*
9D	2005C826	Utility	Expnd_err_msg get_mode init_LEDs clr_ps_cnt
9E	20055586	List_diagnostics	*
9F	20062B32	Create_A0_Script	**********
C1	200546D0	SSC_RAM_Data	*
C2	200548A6	SSC_RAM_Data_Addr	*
C5	20059581	SSC_registers	*
C6	20054614	SSC_powerup	*********
C7	2005967C	SSC_CBTCR_timeout	***

[1]These addresses may change with different versions of the software.

Parameters that you can specify are written out, as shown in the following examples:

```
54  20054BB9 Virtual mode      ******
30  2005D07C MEM_bitmap        *** mark_Hard_SBEs ******
```

The virtual mode test on the first line contains several parameters, but you cannot specify any of them. To run this test individually, enter:

```
>>> T 54
```

The MEM_bitmap test on the second line accepts ten parameters, but you can specify only the fourth one. To mark pages bad in the bitmap for single-bit or multi-bit errors, enter a 1 in the fourth parameter field:

```
>>> T 30 0 0 0 1
```

You must enter a value of either 0 (zero) or 1 (one) for the first three parameters. (0 is used in this example.) The values have no effect on the test; they are simply place holders for the first three parameters. You do not have to specify a value for parameters that follow the user-defined parameter.

4.3.2 Scripts

Most of the tests shown by utility 9E are arranged into scripts. A *script* is a data structure that points to various tests and defines the order in which they are run. Different scripts can run the same set of tests, but in a different order and/or with different parameters and flags. A script also contains the following information:

- The parameters and flags that need to be passed to the test.
- Where the tests can be run from. For example, certain tests can be run only from the EPROM. Other tests are program-independent code, and can be run from EPROM or main memory, to enhance execution speed.
- What is to be shown, if anything, on the console.
- What is to be shown, if anything, in the LED display.
- What action to take on errors (halt, repeat, continue).

The power-up script runs every time the system is powered up. You can also invoke the power-up script at any time by entering T 0.

Additional scripts are included in the ROMs for use in manufacturing and engineering environments. Customer Services personnel can run these scripts and tests individually, using the T command. When doing so, note that certain tests may be dependent upon a state set up from a previous test. For this reason you should use the UNJAM and INITIALIZE commands,

described in Chapter 3, before running an individual test. You do not need to use these commands on system power-up, however, because system power-up leaves the machine in a defined state.

Customer Services personnel with a detailed knowledge of the KA660 hardware and firmware can also create their own scripts, by using the 9F utility. (See Section 4.3.3.) Table 4–2 lists the scripts.

Table 4–2: Scripts Available to Customer Services

Script	Enter with TEST Command
A0	Soft script created by T 9F
A1	Functional verify, usually continue-on-error, with countdown announcements
A3	Functional verify, stop on error, test no. announcements
A4	Loop on A3 functional verify
A5	Address shorts test, run fastest way possible
A6	Memory tests, mark only multi-bit errors
A7	Memory tests; can be run by itself; will continue on error; useful when you want to bypass the bitmap test
A8	Memory acceptance tests; marks single and multi-bit ECC errors in the bitmap; calls A7
A9	Memory tests; halts on the first hard or soft error
B5	SOC cache debug script

4.3.3 User Created Scripts

You can create your own script using utility 9F, to control the order in which tests are run and to select specific parameters and flags for individual tests. In this way you do not have to use the defaults provided by the hardwired scripts.

Utility 9F also provides an easy way to see what flags and parameters are used by the diagnostic executive for each test.

Run test 9F first to build the user script. (See Example 4–1.) Press Return to use the default parameters or flags, which are shown in parentheses. 9F prompts you for the following information:

- Script location. The script can be located in the 1-Kbyte NVRAM in the SSC or in main memory. A script is limited by the size of the data structure that contains it. A larger script can be developed in main memory.
- Test number

- Run environment. This defines where the actual diagnostic test can be run from. The choices are 0 = ROM, 2 = main memory, and 3 = fastest possible. Choose number 3 to select the fastest possible data structure to run from that will not overwrite the test.
- Repeat code
- Error severity level
- Console error report
- Script error treatment
- LED display
- Console display
- Parameters

Example 4–1 shows how to build and run a user script.

The utility displays the test name after you enter the test number, and the number of bytes remaining after you enter the information for each test. When you have finished entering tests, press Return at the next `Next test number:` prompt to end the script building session. Then enter T A0 and press Return to run the new script.

You can review or edit a script you have created.

Example 4–1: Creating a Script with Utility 9F

```
>>>T 9F
SP=201406A8
Script in ?[0=SSC, 2=RAM] :0
Script starts at 20140794
 36 bytes left
Test number (? for list) or script number :80
CQBIC_memory_LMGH>> Run from ?[0=ROM,2=RAM,3=fastest possible] (0):0
CQBIC_memory_LMGH>> Error severity ? [0,1,2,3] (02):0
CQBIC_memory_LMGH>> Console error report? [0=none,1=full] (01):0
CQBIC_memory_LMGH>> Stop script on error? [0=NO,1=YES] (01):0
CQBIC_memory_LMGH>> Repeat? [0=no,1=forever,>1=count<FF] (00):0
CQBIC_memory_LMGH>> LED on entry (01):0
CQBIC_memory_LMGH>> Console Announcement on entry (80):1
 32 bytes left
Test number (? for list) or script number :1
No such diagnostic!
 32 bytes left
Test number (? for list) or script number :
>>>
>>>! Execute test script.

>>>T A0
01..
>>>
```

Example 4–2 shows the script building procedure to follow if (a) you are unsure of the test number to specify, and (b) you want to run one test repeatedly. If you are not sure of the test number, enter ? at the `Next test number:` prompt to invoke test 9E and display test numbers, test names, and so on. To run one test repeatedly enter the following sequence:

1. Enter the test number (40 in Example 4–2) at the `Next test number:` prompt.
2. Enter A0 at the next `Next test number:` prompt.
3. Press Return at the next `Next test number:` prompt.
4. Enter T A0 to begin running the script repeatedly.
5. Press CTRL/C to stop the test.

The above sequence is a useful alternative to using the REPEAT console command to run a test, because REPEAT (test) displays line feeds only; it does not display the console test announcement.

Example 4–2: Listing and Repeating Tests with Utility 9F

```
>>>T 9F
SP=201406A8
Script in ?[0=SSC, 2=RAM] :0
Script starts at 20140794
 36 bytes left
Test number (? for list) or script number :80
CQBIC_memory_LMGH>> Run from ?[0=ROM, 2=RAM, 3=fastest possible] (0):2
CQBIC_memory_LMGH>> Error severity ? [0,1,2,3] (02):2
CQBIC_memory_LMGH>> Console error report? [0=none,1=full] (01):0
CQBIC_memory_LMGH>> Stop script on error? [0=NO,1=YES] (01):0
CQBIC_memory_LMGH>> Repeat? [0=no,1=forever,>1=count<FF] (00):0
CQBIC_memory_LMGH>> LED on entry (01):0
CQBIC_memory_LMGH>> Console Announcement on entry (80):1
 32 bytes left
Test number (? for list) or script number :1
No such diagnostic!
 32 bytes left
Test number (? for list) or script number :
>>>
>>>! Execute test script.

>>>
```

Example 4–3: Console Display (No Errors)

```
KA660-A Vn.n VMB 2.12
Performing normal system tests.
95..94..93..92..91..90..89..88..87..86..85..84..83..82..81..80..
79..78..77..76..75..74..73..72..71..70..69..68..67..66..65..64..
63..62..61..60..59..58..57..56..55..54..53..52..51..50..49..48..
47..46..45..44..43..42..41..40..39..38..37..36..35..34..33..32..
31..30..29..28..27..26..25..24..23..22..21..20..19..18..17..16..
15..14..13..12..11..10..09..08..07..06..05..04..03..
Tests completed.
>>>
```

4.3.4 Console Displays

Example 4–3 shows a typical console display during execution of the ROM-based diagnostics. The numbers on the console display do not refer to actual test numbers. Refer to Table 4–5 to see the correspondence between the numbers displayed (listed in the Console Display column) and the actual tests being run (listed in the Test column).

The first line contains the firmware revision and the virtual memory bootstrap (VMB) revision.

Diagnostic test failures, if specified in the firmware script, produce an error display in the format shown in Example 4–4.

Example 4–4: Sample Output with Errors

```
?45 2 10 FF 0000 0002 00; SUBTEST_45_10, DE_SOC_Cache_mem_CQBIC.LIS

P1=00000010  P2=00000000  P3=04000000  P4=04000000  P5=00000015
P6=20089000  P7=00000001  P8=80000400  P9=00000010 P10=00000000
r0=00000000  r1=00000000  r2=43214321  r3=30080000  r4=00000400
r5=20089000  r6=43214321  r7=43214321  r8=00000000 EPC=00000000

Tests completed
```

The errors are printed in a five-line display. The first line has seven fields:

```
Test Severity Subtestlog Error_type Vector Count Loop_subtestlog
```

- Test identifies the diagnostic test.
- Severity is the severity level of a test failure, as dictated by the script. Failure of a severity level 2 (SV2) test causes the display of this five-line error printout, and halts an autoboot. An error of severity level 1 causes a display of the first line of the error printout, but does not interrupt an autoboot. Most tests have a severity level of 2.
- Subtestlog is two hexadecimal digits identifying, usually within 10 instructions, where in the diagnostic the error occurred.
- Error_type signals the diagnostic's state and any illegal behavior. This field indicates a condition that the diagnostic expects on detecting a failure. FE or EF in this field means that an unexpected exception or interrupt was detected. FF indicates an error as a result of normal testing, such as a miscompare. The possible codes are:

  FF—Normal error exit from diagnostic
  FE—Unanticipated interrupt
  FD—Interrupt in cleanup routine
  FC—Interrupt in interrupt handler
  FB—Script requirements not met
  FA—No such diagnostic
  EF—Unanticipated exception in executive

- Vector identifies the SCB vector (0000 in the example above) through which the unexpected exception or interrupt trapped, when the error_type field detects an unexpected exception or interrupt (FE, FD, FC, or EF).
- Count is four hexadecimal digits. It shows the number of previous errors that have occurred (two in Example 4–4).
- Loop_subtestlog is used for calling out routines that identify specific errors.

Lines 2 and 3 of the error printout are parameters 1 through 10. When the diagnostics are running normally, these parameters are the same parameters that are listed in Table 4–1.

When an unexpected machine check exception or other type of exception occurs during the executive (error_type is EF), the stack is saved in the parameters on lines 2 and 3, as listed in Tables 4–3 and 4–4.

Table 4–3: Values Saved, Machine Check Exception During EF

Parameter	Value
P1	Contents of SP, points to vector value in P2
P2	Vector = 04, vector of exception 04–3FC
P3	Address of PC pointing to failed instruction, P9
P4	Byte count = 10
P5	Machine check code
P6	Most recent virtual address
P7	Internal state information 1
P8	Internal state information 2
P9	PC, points to failing instruction
P10	PSL

Table 4–4: Values Saved, Exception During Executive

Parameter	Value
P1	Contents of SP, points to vector value in P2
P2	Vector = nn, vector of exception 04–3FC
P3	Address of PC pointing to failed instruction, P4
P4	PC, points to instruction following failed instruction
P5	PSL
P6	Contents of stack
P7	Contents of stack
P8	Contents of stack
P9	Contents of stack
P10	Contents of stack

Lines 4 and 5 of the error printout are general registers R0 through R8 and the exception PC (if it occurred).

When returning a module for repair, record the first line of the error printout and the version of the ROMs on the module repair tag.

The Default Action on Error column refers to the action taken by the diagnostic executive under the following circumstances:

- The diagnostic executive detects an unexpected exception or interrupt.
- A test fails and that failure is reported to the diagnostic executive.

The Default on Error column does not refer to the action taken by the memory tests. The diagnostic executive either halts the script or continues execution at the next test in the script.

Most memory tests have a continue-on-error parameter (labeled cont_on_error, as shown in test 47 in Example 4–2). If you explicitly set cont_on_error using parameter 4 in a memory test, the test marks bad pages in the bitmap and continues without notifying the diagnostic executive of the error. In this case, a halt on error does not occur even if you specify halt on error in the diagnostic executive (by answering Yes to `Stop script on error?` in Utility 9F), since the memory test does not notify the diagnostic executive that an error has occurred.

Figure 4–1 shows the LEDs on the KA660 CPU. They correspond to the hexadecimal display on the CPU cover panel.

Figure 4–1: KA660 CPU Module LEDs

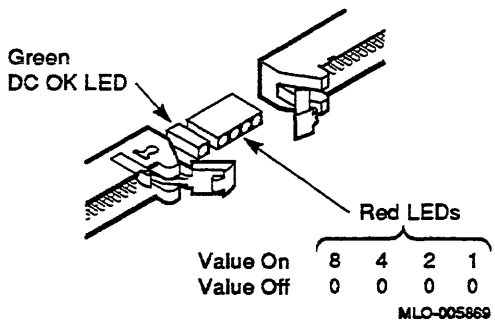


Table 4–5: KA660 Console Displays and FRU Pointers

No.	Test	LED	Description	FRUs	Conditions
script_A1:					
95	9D	C	Utility	1	RPE-CON-SV2-VOF-RHP-ROM
94	42	B	check_for_intrs	1, 4	RPE-CON-SV2-VOF-RHP-ROM
93	33	8	CMCTL_chk_init	1	RPE-CON-SV2-VOF-RHP-ROM
92	32	8	CMCTL_registers	1	RPE-CON-SV2-VOF-RHP-ROM
91	31	8	CSR_setup	1, 2	RPE-CON-SV2-VOF-RHP-ROM
90	30	8	map_setup	1, 2	RPE-STP-SV2-VOF-RHP-ROM
89	54	B	virtual	1	RPE-CON-SV2-VOF-RHP-ROM
88	49	8	memory_test_fdm	2, 1, 3	RPE-CON-SV2-VOF-RHP-ROM
87	60	6	serial_line	1, 6	RPE-CON-SV2-VOF-RHP-ROM
85	90	7	registers	1, 4, 3	RPE-CON-SV2-VOF-RHP-ROM
84	C6	C	CSSC_chk_init	1	RPE-CON-SV2-VOF-RHP-ROM
83	52	C	PROG_TIME	1	RPE-CON-SV2-VOF-RHP-ROM
82	52	C	PROG_TIME	1	RPE-CON-SV2-VOF-RHP-ROM
81	53	C	TOY	7, 1	RPE-CON-SV2-VOF-RHP-ROM
80	C1	C	SSC_RAM	1	RPE-CON-SV2-VOF-RHP-ROM
79	34	C	ROM_logic	1	RPE-CON-SV2-VOF-RHP-ROM
78	C5	C	SSC_registers	1	RPE-CON-SV2-VOF-RHP-ROM
76	C7	C	CBTCR_timeout	1	RPE-CON-SV2-VOF-RHP-ROM
75	46	B	SOC_cache_diag_mode	1	NER-CON-SV1-VOF-RHU-ROM
74	46	B	SOC_cache_diag_mode	1	NER-CON-SV1-VOF-RHU-ROM
73	46	B	SOC_cache_diag_mode	1	NER-CON-SV1-VOF-RHU-ROM
72	46	B	SOC_cache_diag_mode	1	NER-CON-SV1-VOF-RHU-ROM
71	46	B	SOC_cache_diag_mode	1	NER-CON-SV1-VOF-RHU-ROM
70	46	B	SOC_cache_diag_mode	1	NER-CON-SV1-VOF-RHU-ROM
69	46	B	SOC_cache_diag_mode	1	NER-CON-SV1-VOF-RHU-ROM
68	46	B	SOC_cache_diag_mode	1	NER-CON-SV1-VOF-RHU-ROM
67	44	B	SOC_D_Cache_w_memory	1	NER-CON-SV1-VOF-RHU-ROM

Field-replaceable Units:
FRU 1: KA660; FRU 2: MS650; FRU 3: Backplane; FRU 4: Q22-Bus Device; FRU 5: System Power Supply; FRU 6: CPU Cover Panel; FRU 7: Battery

Conditions:
NER-0, RPE-1 : Report error
CON-0, STP-1 : Action on error
SV1-1, SV2-2 : Severity level
VOF-0, VON-1 : Virtual mode
RHP-0, RHU-1 : Halt protection
ROM-0, RAM-2, FAST-3 : Run environment

Table 4–5 (Cont.): KA660 Console Displays and FRU Pointers

No.	Test	LED	Description	FRUs	Conditions
script_A1:					
66	4F	8	memory_data	2, 1, 3	RPE-CON-SV2-VOF-RHP-ROM
65	4E	8	memory_byte	2, 1, 3	RPE-CON-SV2-VOF-RHP-FAST
64	4D	8	memory_addr	2, 1, 3	RPE-CON-SV2-VOF-RHP-FAST
63	4C	8	memory_ECC_error	2, 1, 3	RPE-CON-SV2-VOF-RHP-FAST
62	4B	8	mask_write_w_errs	2, 1, 3	RPE-CON-SV2-VOF-RHP-FAST
61	4A	8	ECC_correction	2, 1	RPE-CON-SV2-VOF-RHP-FAST
60	3F	8	mem_FDM_addr_shorts	2, 1, 3	RPE-CON-SV2-VOF-RHP-FAST
59	48	8	addr_shrts	2, 1, 3	RPE-CON-SV2-VOF-RHP-FAST
58	48	8	addr_shrts	2, 1, 3	RPE-CON-SV2-VOF-RHP-FAST
57	48	8	addr_shrts	2, 1, 3	RPE-CON-SV2-VOF-RHP-FAST
56	48	8	addr_shrts	2, 1, 3	RPE-CON-SV2-VOF-RHP-FAST
55	48	8	addr_shrts	2, 1, 3	RPE-CON-SV2-VOF-RHP-FAST
54	48	8	addr_shrts	2, 1, 3	RPE-CON-SV2-VOF-RHP-FAST
53	48	8	addr_shrts	2, 1, 3	RPE-CON-SV2-VOF-RHP-FAST
52	48	8	addr_shrts	2, 1, 3	RPE-CON-SV2-VOF-RHP-FAST
51	48	8	addr_shrts	2, 1, 3	RPE-CON-SV2-VOF-RHP-FAST
50	48	8	addr_shrts	2, 1, 3	RPE-CON-SV2-VOF-RHP-FAST
48	48	8	addr_shrts	2, 1, 3	RPE-CON-SV2-VOF-RHP-FAST
47	48	8	addr_shrts	2, 1, 3	RPE-CON-SV2-VOF-RHP-FAST
46	48	8	addr_shrts	2, 1, 3	RPE-CON-SV2-VOF-RHP-FAST
45	48	8	addr_shrts	2, 1, 3	RPE-CON-SV2-VOF-RHP-FAST
44	48	8	addr_shrts	2, 1, 3	RPE-CON-SV2-VOF-RHP-FAST
43	47	8	memory_refresh	2, 1	RPE-CON-SV2-VOF-RHP-FAST
42	40	8	count_bad_pages	2	RPE-CON-SV1-VOF-RHP-ROM
40	44	B	SOC_D_Cache_w_memory	1, 2	NER-CON-SV1-VOF-RHU-ROM
39	44	B	SOC_D_Cache_w_memory	1, 2	NER-CON-SV1-VOF-RHU-ROM
38	44	B	SOC_D_Cache_w_memory	1, 2	NER-CON-SV1-VOF-RHU-ROM
37	44	B	SOC_D_Cache_w_memory	1, 2	NER-CON-SV1-VOF-RHU-ROM

Field-replaceable Units:
FRU 1: KA660; FRU 2: MS650; FRU 3: Backplane; FRU 4: Q22-Bus Device; FRU 5: System Power Supply; FRU 6: CPU Cover Panel; FRU 7: Battery

Conditions:
NER-0, RPE-1 : Report error
CON-0, STP-1 : Action on error
SV1-1, SV2-2 : Severity level
VOF-0, VON-1 : Virtual mode
RHP-0, RHU-1 : Halt protection
ROM-0, RAM-2, FAST-3 : Run environment

Table 4–5 (Cont.): KA660 Console Displays and FRU Pointers

No.	Test	LED	Description	FRUs	Conditions
script_A1:					
36	44	B	SOC_D_Cache_w_memory	1, 2	NER-CON-SV1-VOF-RHU-ROM
35	44	B	SOC_D_Cache_w_memory	1, 2	NER-CON-SV1-VOF-RHU-ROM
34	44	B	SOC_D_Cache_w_memory	1, 2	NER-CON-SV1-VOF-RHU-ROM
33	C2	C	SSC_RAM_addr_shrts	1	RPE-CON-SV2-VOF-RHP-ROM
32	80	7	CQBIC_memory	1, 2	RPE-CON-SV2-VOF-RHP-FAST
31	45	7	cache_mem_cqbic	1, 2	NER-CON-SV1-VOF-RHU-ROM
30	45	7	cache_mem_cqbic	1, 2	NER-CON-SV1-VOF-RHU-ROM
29	45	7	cache_mem_cqbic	1, 2	NER-CON-SV1-VOF-RHU-ROM
27	45	7	cache_mem_cqbic	1, 2	NER-CON-SV1-VOF-RHU-ROM
26	45	7	cache_mem_cqbic	1, 2	NER-CON-SV1-VOF-RHU-ROM
24	45	7	cache_mem_cqbic	1, 2	NER-CON-SV1-VOF-RHU-ROM
23	43	B	SOC_DI_Cache_w_memory	1, 2	NER-CON-SV1-VOF-RHU-FAST
21	43	B	SOC_DI_Cache_w_memory	1, 2	NER-CON-SV1-VOF-RHU-FAST
20	43	B	SOC_DI_Cache_w_memory	1, 2	NER-CON-SV1-VOF-RHU-FAST
19	43	B	SOC_DI_Cache_w_memory	1, 2	NER-CON-SV1-VOF-RHU-FAST
18	43	B	SOC_DI_Cache_w_memory	1, 2	NER-CON-SV1-VOF-RHU-FAST
17	43	B	SOC_DI_Cache_w_memory	1, 2	NER-CON-SV1-VOF-RHU-FAST
16	43	B	SOC_DI_Cache_w_memory	1, 2	NER-CON-SV1-VOF-RHU-FAST
15	43	B	SOC_DI_Cache_w_memory	1, 2	NER-CON-SV1-VOF-RHU-ROM
14	43	B	SOC_DI_Cache_w_memory	1, 2	NER-CON-SV1-VOF-RHU-FAST
13	5A	8	SOC_CMCTL	1, 2	RPE-CON-SV2-VOF-RHP-RAM
12	51	A	FPA	1	RPE-CON-SV2-VOF-RHP-FAST
11	5F	4	SGEC_func	1, 6	RPE-CON-SV2-VOF-RHP-ROM
10	5C	5	SHAC_func	1, 3	RPE-CON-SV2-VOF-RHP-ROM
09	9A	8	Interaction_func	1, 2, 3	RPE-CON-SV2-VOF-RHP-FAST
08	83	7	qza_lpbck1	4	RPE-CON-SV2-VOF-RHP-ROM
07	84	7	qza_lpbck2	4	RPE-CON-SV2-VOF-RHP-ROM
06	85	7	qza_memory	4	RPE-CON-SV2-VOF-RHP-ROM

Field-replaceable Units:
FRU 1: KA660; FRU 2: MS650; FRU 3: Backplane; FRU 4: Q22-Bus Device; FRU 5: System Power Supply; FRU 6: CPU Cover Panel; FRU 7: Battery

Conditions:
NER-0, RPE-1 : Report error
CON-0, STP-1 : Action on error
SV1-1, SV2-2 : Severity level
VOF-0, VON-1 : Virtual mode
RHP-0, RHU-1 : Halt protection
ROM-0, RAM-2, FAST-3 : Run environment

Table 4–5 (Cont.): KA660 Console Displays and FRU Pointers

No.	Test	LED	Description	FRUs	Conditions
script_A1:					
05	86	7	qza_dma	4	RPE-CON-SV2-VOF-RHP-ROM
04	99	B	flush_ena_caches	4	RPE-CON-SV2-VOF-RHP-ROM
03	41	C	board_reset	1, 4	RPE-CON-SV2-VOF-RHP-ROM
script_A2:					
9D	9D	C	Utility	1, 2	RPE-STP-SV2-VOF-RHP-ROM
42	42	B	check_for_intrs	1, 4	RPE-STP-SV2-VOF-RHP-ROM
C6	C6	C	CSSC_chk_init	1	RPE-STP-SV2-VOF-RHP-ROM
60	60	6	Serial_line	1, 6	RPE-STP-SV2-VOF-RHP-ROM
52	52	C	PROG_TIME	1	RPE-STP-SV2-VOF-RHP-ROM
52	52	C	PROG_TIME	1	RPE-STP-SV2-VOF-RHP-ROM
53	53	C	TOY	7, 1	RPE-STP-SV2-VOF-RHP-ROM
C1	C1	C	SSC_RAM	1	RPE-STP-SV2-VOF-RHP-ROM
34	34	C	ROM_logic	1	RPE-STP-SV2-VOF-RHP-ROM
91	91	7	CQBIC_chk_init	1, 4, 3	RPE-STP-SV2-VOF-RHP-ROM
C5	C5	C	SSC_registers	1	RPE-STP-SV2-VOF-RHP-ROM
55	55	B	interval_timer	1	RPE-STP-SV2-VOF-RHP-ROM
46	46	B	SOC_cache_diag_mode	1	NER-CON-SV1-VOF-RHU-ROM
46	46	B	SOC_cache_diag_mode	1	NER-CON-SV1-VOF-RHU-ROM
46	46	B	SOC_cache_diag_mode	1	NER-CON-SV1-VOF-RHU-ROM
46	46	B	SOC_cache_diag_mode	1	NER-CON-SV1-VOF-RHU-ROM
46	46	B	SOC_cache_diag_mode	1	NER-CON-SV1-VOF-RHU-ROM
46	46	B	SOC_cache_diag_mode	1	NER-CON-SV1-VOF-RHU-ROM
46	46	B	SOC_cache_diag_mode	1	NER-CON-SV1-VOF-RHU-ROM
46	46	B	SOC_cache_diag_mode	1	NER-CON-SV1-VOF-RHU-ROM
99	99	B	flush_ena_caches	1	RPE-STP-SV2-VOF-RHP-ROM

Field-replaceable Units:
FRU 1: KA660; FRU 2: MS650; FRU 3: Backplane; FRU 4: Q22-Bus Device; FRU 5: System Power Supply; FRU 6: CPU Cover Panel; FRU 7: Battery

Conditions:
NER-0, RPE-1 : Report error
CON-0, STP-1 : Action on error
SV1-1, SV2-2 : Severity level
VOF-0, VON-1 : Virtual mode
RHP-0, RHU-1 : Halt protection
ROM-0, RAM-2, FAST-3 : Run environment

Table 4–5 (Cont.): KA660 Console Displays and FRU Pointers

No.	Test	LED	Description	FRUs	Conditions
script_A2:					
90	90	7	registers	1, 4, 3	RPE-STP-SV2-VOF-RHP-ROM
32	32	8	CMCTL_registers	1, 2	RPE-STP-SV2-VOF-RHP-ROM
C7	C7	C	CBTCR_timeout	1	RPE-STP-SV2-VOF-RHP-ROM
5C	5C	5	SHAC_func	1, 3	RPE-STP-SV2-VOF-RHP-ROM
script_A3:					
9D	9D	C	Utility	1	RPE-STP-SV2-VOF-RHP-ROM
42	42	B	check_for_intrs	1, 4	RPE-STP-SV2-VOF-RHP-ROM
33	33	8	CMCTL_chk_init	1	RPE-STP-SV2-VOF-RHP-ROM
31	31	8	CSR_setup	1, 2	RPE-STP-SV2-VOF-RHP-ROM
30	30	8	map_setup	2, 1	RPE-STP-SV2-VOF-RHP-ROM
54	54	B	virtual	1	RPE-STP-SV2-VOF-RHP-ROM
49	49	8	memory_test_fdm	2, 1, 3	RPE-STP-SV2-VOF-RHP-ROM
60	60	6	Serial_line	1, 6	RPE-STP-SV2-VOF-RHP-ROM
91	91	7	CQBIC_chk_init	1, 4, 3	RPE-STP-SV2-VOF-RHP-ROM
90	90	7	registers	1, 4, 3	RPE-STP-SV2-VOF-RHP-ROM
C6	C6	C	CSSC_chk_init	1	RPE-STP-SV2-VOF-RHP-ROM
52	52	C	PROG_TIME	1	RPE-STP-SV2-VOF-RHP-ROM
52	52	C	PROG_TIME	1	RPE-STP-SV2-VOF-RHP-ROM
53	53	C	TOY	7, 1	RPE-STP-SV2-VOF-RHP-ROM
C1	C1	C	SSC_RAM	1	RPE-STP-SV2-VOF-RHP-ROM
C5	C5	C	SSC_registers	1	RPE-STP-SV2-VOF-RHP-ROM
55	55	B	interval_timer	1	RPE-STP-SV2-VOF-RHP-ROM
C7	C7	C	CBTCR_timeout	1	RPE-STP-SV2-VOF-RHP-ROM
46	46	B	SOC_cache_diag_mode	1	NER-CON-SV1-VOF-RHU-ROM
46	46	B	SOC_cache_diag_mode	1	NER-CON-SV1-VOF-RHU-ROM

Field-replaceable Units:
FRU 1: KA660; FRU 2: MS650; FRU 3: Backplane; FRU 4: Q22-Bus Device; FRU 5: System Power Supply; FRU 6: CPU Cover Panel; FRU 7: Battery

Conditions:
NER-0, RPE-1 : Report error
CON-0, STP-1 : Action on error
SV1-1, SV2-2 : Severity level
VOF-0, VON-1 : Virtual mode
RHP-0, RHU-1 : Halt protection
ROM-0, RAM-2, FAST-3 : Run environment

Table 4–5 (Cont.): KA660 Console Displays and FRU Pointers

No.	Test	LED	Description	FRUs	Conditions
script_A3:					
46	46	B	SOC_cache_diag_mode	1	NER-CON-SV1-VOF-RHU-ROM
46	46	B	SOC_cache_diag_mode	1	NER-CON-SV1-VOF-RHU-ROM
46	46	B	SOC_cache_diag_mode	1	NER-CON-SV1-VOF-RHU-ROM
46	46	B	SOC_cache_diag_mode	1	NER-CON-SV1-VOF-RHU-ROM
46	46	B	SOC_cache_diag_mode	1	NER-CON-SV1-VOF-RHU-ROM
46	46	B	SOC_cache_diag_mode	1	NER-CON-SV1-VOF-RHU-ROM
44	44	B	SOC_D_Cache_w_memory	1, 2	NER-CON-SV1-VOF-RHU-ROM
44	44	B	SOC_D_Cache_w_memory	1, 2	NER-CON-SV1-VOF-RHU-ROM
44	44	B	SOC_D_Cache_w_memory	1, 2	NER-CON-SV1-VOF-RHU-ROM
44	44	B	SOC_D_Cache_w_memory	1, 2	NER-CON-SV1-VOF-RHU-ROM
44	44	B	SOC_D_Cache_w_memory	1, 2	NER-CON-SV1-VOF-RHU-ROM
44	44	B	SOC_D_Cache_w_memory	1, 2	NER-CON-SV1-VOF-RHU-ROM
44	44	B	SOC_D_Cache_w_memory	1, 2	NER-CON-SV1-VOF-RHU-ROM
44	44	B	SOC_D_Cache_w_memory	1, 2	NER-CON-SV1-VOF-RHU-ROM
4F	4F	8	memory_data	2, 1, 3	RPE-STP-SV2-VOF-RHP-ROM
4E	4E	8	memory_byte	2, 1, 3	RPE-STP-SV2-VOF-RHP-FAST
4D	4D	8	memory_addr	2, 1, 3	RPE-STP-SV2-VOF-RHP-FAST
4C	4C	8	memory_ECC_error	2, 1, 3	RPE-STP-SV2-VOF-RHP-FAST
4B	4B	8	mask_write_w_errs	2, 1, 3	RPE-STP-SV2-VOF-RHP-FAST
4A	4A	8	ECC_correction	2, 1	RPE-STP-SV2-VOF-RHP-FAST
3F	3F	8	mem_FDM_addr_shorts	2, 1, 3	RPE-STP-SV2-VOF-RHP-FAST
48	48	8	addr_shrts	2, 1, 3	RPE-STP-SV2-VOF-RHP-FAST
48	48	8	addr_shrts	2, 1, 3	RPE-STP-SV2-VOF-RHP-FAST
48	48	8	addr_shrts	2, 1, 3	RPE-STP-SV2-VOF-RHP-FAST
48	48	8	addr_shrts	2, 1, 3	RPE-STP-SV2-VOF-RHP-FAST
48	48	8	addr_shrts	2, 1, 3	RPE-STP-SV2-VOF-RHP-FAST
48	48	8	addr_shrts	2, 1, 3	RPE-STP-SV2-VOF-RHP-FAST
48	48	8	addr_shrts	2, 1, 3	RPE-STP-SV2-VOF-RHP-FAST

Field-replaceable Units:
FRU 1: KA660; FRU 2: MS650; FRU 3: Backplane; FRU 4: Q22-Bus Device; FRU 5: System Power Supply; FRU 6: CPU Cover Panel; FRU 7: Battery

Conditions:
NER-0, RPE-1 : Report error
CON-0, STP-1 : Action on error
SV1-1, SV2-2 : Severity level
VOF-0, VON-1 : Virtual mode
RHP-0, RHU-1 : Halt protection
ROM-0, RAM-2, FAST-3 : Run environment

Table 4–5 (Cont.): KA660 Console Displays and FRU Pointers

No.	Test	LED	Description	FRUs	Conditions
script_A3:					
48	48	8	addr_shrts	2, 1, 3	RPE-STP-SV2-VOF-RHP-FAST
48	48	8	addr_shrts	2, 1, 3	RPE-STP-SV2-VOF-RHP-FAST
48	48	8	addr_shrts	2, 1, 3	RPE-STP-SV2-VOF-RHP-FAST
48	48	8	addr_shrts	2, 1, 3	RPE-STP-SV2-VOF-RHP-FAST
48	48	8	addr_shrts	2, 1, 3	RPE-STP-SV2-VOF-RHP-FAST
48	48	8	addr_shrts	2, 1, 3	RPE-STP-SV2-VOF-RHP-FAST
48	48	8	addr_shrts	2, 1, 3	RPE-STP-SV2-VOF-RHP-FAST
48	48	8	addr_shrts	2, 1, 3	RPE-STP-SV2-VOF-RHP-FAST
47	47	8	memory_refresh	2, 1	RPE-STP-SV2-VOF-RHP-FAST
40	40	8	count_bad_pages	2	RPE-STP-SV2-VOF-RHP-ROM
44	44	B	SOC_D_Cache_w_memory	1, 2	NER-CON-SV1-VOF-RHU-ROM
44	44	B	SOC_D_Cache_w_memory	1, 2	NER-CON-SV1-VOF-RHU-ROM
44	44	B	SOC_D_Cache_w_memory	1, 2	NER-CON-SV1-VOF-RHU-ROM
44	44	B	SOC_D_Cache_w_memory	1, 2	NER-CON-SV1-VOF-RHU-ROM
44	44	B	SOC_D_Cache_w_memory	1, 2	NER-CON-SV1-VOF-RHU-ROM
44	44	B	SOC_D_Cache_w_memory	1, 2	NER-CON-SV1-VOF-RHU-ROM
44	44	B	SOC_D_Cache_w_memory	1, 2	NER-CON-SV1-VOF-RHU-ROM
44	44	B	SOC_D_Cache_w_memory	1, 2	NER-CON-SV1-VOF-RHU-ROM
80	80	7	CQBIC_memory	1, 2	RPE-STP-SV2-VOF-RHP-FAST
45	45	7	cache_mem_cqbic	1, 2	NER-CON-SV1-VOF-RHU-ROM
45	45	7	cache_mem_cqbic	1, 2	NER-CON-SV1-VOF-RHU-ROM
45	45	7	cache_mem_cqbic	1, 2	NER-CON-SV1-VOF-RHU-ROM
45	45	7	cache_mem_cqbic	1, 2	NER-CON-SV1-VOF-RHU-ROM
45	45	7	cache_mem_cqbic	1, 2	NER-CON-SV1-VOF-RHU-ROM
45	45	7	cache_mem_cqbic	1, 2	NER-CON-SV1-VOF-RHU-ROM
45	45	7	cache_mem_cqbic	1, 2	NER-CON-SV1-VOF-RHU-ROM
45	45	7	cache_mem_cqbic	1, 2	NER-CON-SV1-VOF-RHU-ROM
43	43	B	SOC_DI_Cache_w_memory	1, 2	NER-CON-SV1-VOF-RHU-FAST

Field-replaceable Units:
FRU 1: KA660; FRU 2: MS650; FRU 3: Backplane; FRU 4: Q22-Bus Device; FRU 5: System Power Supply; FRU 6: CPU Cover Panel; FRU 7: Battery

Conditions:
NER-0, RPE-1 : Report error
CON-0, STP-1 : Action on error
SV1-1, SV2-2 : Severity level
VOF-0, VON-1 : Virtual mode
RHP-0, RHU-1 : Halt protection
ROM-0, RAM-2, FAST-3 : Run environment

Table 4–5 (Cont.): KA660 Console Displays and FRU Pointers

No.	Test	LED	Description	FRUs	Conditions
script_A3:					
43	43	B	SOC_DI_Cache_w_memory	1, 2	NER-CON-SV1-VOF-RHU-FAST
43	43	B	SOC_DI_Cache_w_memory	1, 2	NER-CON-SV1-VOF-RHU-FAST
43	43	B	SOC_DI_Cache_w_memory	1, 2	NER-CON-SV1-VOF-RHU-FAST
43	43	B	SOC_DI_Cache_w_memory	1, 2	NER-CON-SV1-VOF-RHU-FAST
43	43	B	SOC_DI_Cache_w_memory	1, 2	NER-CON-SV1-VOF-RHU-FAST
43	43	B	SOC_DI_Cache_w_memory	1, 2	NER-CON-SV1-VOF-RHU-FAST
43	43	B	SOC_DI_Cache_w_memory	1, 2	NER-CON-SV1-VOF-RHU-FAST
43	43	B	SOC_DI_Cache_w_memory	1, 2	NER-CON-SV1-VOF-RHU-ROM
43	43	B	SOC_DI_Cache_w_memory	1, 2	NER-CON-SV1-VOF-RHU-FAST
5A	5A	8	SOC_CMCTL	1, 2	RPE-STP-SV2-VOF-RHU-RAM
51	51	A	FPA	1	RPE-STP-SV2-VOF-RHP-FAST
5F	5F	4	SGEC_func	1, 6	RPE-STP-SV2-VOF-RHP-ROM
5C	5C	5	SHAC_func	1, 3	RPE-STP-SV2-VOF-RHP-ROM
9A	9A	8	Interaction_func	1, 2	RPE-STP-SV2-VOF-RHP-FAST
99	99	B	flush_ena_caches	1, 2	RPE-STP-SV2-VOF-RHP-ROM
41	41	C	board_reset	2, 1, 3	RPE-STP-SV2-VOF-RHP-ROM
9D	9D	C	Utility	1	RPE-STP-SV2-VOF-RHP-ROM
script_A5:					
3F	3F	8	mem_FDM_addr_shorts	2, 1, 3	RPE-CON-SV2-VOF-RHU-FAST
48	48	8	addr_shrts	2, 1, 3	RPE-STP-SV2-VOF-RHP-FAST
48	48	8	addr_shrts	2, 1, 3	RPE-STP-SV2-VOF-RHP-FAST
48	48	8	addr_shrts	2, 1, 3	RPE-STP-SV2-VOF-RHP-FAST
48	48	8	addr_shrts	2, 1, 3	RPE-STP-SV2-VOF-RHP-FAST
48	48	8	addr_shrts	2, 1, 3	RPE-STP-SV2-VOF-RHP-FAST
48	48	8	addr_shrts	2, 1, 3	RPE-STP-SV2-VOF-RHP-FAST

Field-replaceable Units:
FRU 1: KA660; FRU 2: MS650; FRU 3: Backplane; FRU 4: Q22-Bus Device; FRU 5: System Power Supply; FRU 6: CPU Cover Panel; FRU 7: Battery

Conditions:
NER-0, RPE-1 : Report error
CON-0, STP-1 : Action on error
SV1-1, SV2-2 : Severity level
VOF-0, VON-1 : Virtual mode
RHP-0, RHU-1 : Halt protection
ROM-0, RAM-2, FAST-3 : Run environment

Table 4–5 (Cont.): KA660 Console Displays and FRU Pointers

No.	Test	LED	Description	FRUs	Conditions
script_A5:					
48	48	8	addr_shrts	2, 1, 3	RPE-STP-SV2-VOF-RHP-FAST
48	48	8	addr_shrts	2, 1, 3	RPE-STP-SV2-VOF-RHP-FAST
48	48	8	addr_shrts	2, 1, 3	RPE-STP-SV2-VOF-RHP-FAST
48	48	8	addr_shrts	2, 1, 3	RPE-STP-SV2-VOF-RHP-FAST
48	48	8	addr_shrts	2, 1, 3	RPE-STP-SV2-VOF-RHP-FAST
48	48	8	addr_shrts	2, 1, 3	RPE-STP-SV2-VOF-RHP-FAST
48	48	8	addr_shrts	2, 1, 3	RPE-STP-SV2-VOF-RHP-FAST
48	48	8	addr_shrts	2, 1, 3	RPE-STP-SV2-VOF-RHP-FAST
script_A6:					
30	30	8	map_setup	1, 2	RPE-STP-SV2-VOF-RHP-ROM
4F	4F	8	memory_data	2, 1, 3	RPE-STP-SV2-VOF-RHP-ROM
4D	4D	8	memory_addr	2, 1, 3	RPE-STP-SV2-VOF-RHP-FAST
4C	4C	8	memory_ECC_error	2, 1, 3	RPE-STP-SV2-VOF-RHP-FAST
4B	4B	8	mask_write_w_errs	2, 1, 3	RPE-STP-SV2-VOF-RHP-FAST
4A	4A	8	ECC_correction	2, 1, 3	RPE-STP-SV2-VOF-RHP-FAST
3F	3F	8	mem_FDM_addr_shorts	2, 1	RPE-STP-SV2-VOF-RHP-FAST
48	48	8	addr_shrts	2, 1, 3	RPE-STP-SV2-VOF-RHP-FAST
47	47	8	memory_refresh	2, 1	RPE-STP-SV2-VOF-RHP-FAST
40	40	8	count_bad_pages	2	RPE-STP-SV2-VOF-RHP-ROM
80	80	7	CQBIC_memory	1, 2	RPE-STP-SV2-VOF-RHP-FAST
script_A7:					
4F	4F	8	memory_data	2, 1, 3	RPE-STP-SV2-VOF-RHP-ROM
4E	4E	8	memory_byte	2, 1, 3	RPE-STP-SV2-VOF-RHP-FAST

Field-replaceable Units:
FRU 1: KA660; FRU 2: MS650; FRU 3: Backplane; FRU 4: Q22-Bus Device; FRU 5: System Power Supply; FRU 6: CPU Cover Panel; FRU 7: Battery

Conditions:
NER-0, RPE-1 : Report error
CON-0, STP-1 : Action on error
SV1-1, SV2-2 : Severity level
VOF-0, VON-1 : Virtual mode
RHP-0, RHU-1 : Halt protection
ROM-0, RAM-2, FAST-3 : Run environment

Table 4–5 (Cont.): KA660 Console Displays and FRU Pointers

No.	Test	LED	Description	FRUs	Conditions
script_A7:					
4D	4D	8	memory_byte	2, 1, 3	RPE-STP-SV2-VOF-RHP-FAST
4C	4C	8	memory_ECC_error	2, 1, 3	RPE-STP-SV2-VOF-RHP-FAST
4B	4B	8	mask_write_w_errs	2, 1, 3	RPE-STP-SV2-VOF-RHP-FAST
4A	4A	8	ECC_correction	2, 1	RPE-STP-SV2-VOF-RHP-FAST
3F	3F	8	mem_FDM_addr_shorts	2, 1, 3	RPE-STP-SV2-VOF-RHP-FAST
48	48	8	addr_shrts	2, 1, 3	RPE-STP-SV2-VOF-RHP-FAST
48	48	8	addr_shrts	2, 1, 3	RPE-STP-SV2-VOF-RHP-FAST
48	48	8	addr_shrts	2, 1, 3	RPE-STP-SV2-VOF-RHP-FAST
48	48	8	addr_shrts	2, 1, 3	RPE-STP-SV2-VOF-RHP-FAST
48	48	8	addr_shrts	2, 1, 3	RPE-STP-SV2-VOF-RHP-FAST
48	48	8	addr_shrts	2, 1, 3	RPE-STP-SV2-VOF-RHP-FAST
48	48	8	addr_shrts	2, 1, 3	RPE-STP-SV2-VOF-RHP-FAST
48	48	8	addr_shrts	2, 1, 3	RPE-STP-SV2-VOF-RHP-FAST
48	48	8	addr_shrts	2, 1, 3	RPE-STP-SV2-VOF-RHP-FAST
48	48	8	addr_shrts	2, 1, 3	RPE-STP-SV2-VOF-RHP-FAST
48	48	8	addr_shrts	2, 1, 3	RPE-STP-SV2-VOF-RHP-FAST
48	48	8	addr_shrts	2, 1, 3	RPE-STP-SV2-VOF-RHP-FAST
48	48	8	addr_shrts	2, 1, 3	RPE-STP-SV2-VOF-RHP-FAST
48	48	8	addr_shrts	2, 1, 3	RPE-STP-SV2-VOF-RHP-FAST
47	47	8	memory_refresh	2, 1	RPE-STP-SV2-VOF-RHP-FAST
40	40	8	count_bad_pages	2	RPE-CON-SV2-VOF-RHU-ROM
80	80	7	CQBIC_memory	1, 2	RPE-CON-SV2-VOF-RHU-FAST
41	41	C	board_reset	2, 1, 3	RPE-STP-SV2-VOF-RHP-ROM
script_A8:					
31	31	8	CSR_setup	1, 2	RPE-STP-SV2-VOF-RHP-ROM

Field-replaceable Units:
FRU 1: KA660; FRU 2: MS650; FRU 3: Backplane; FRU 4: Q22-Bus Device; FRU 5: System Power Supply; FRU 6: CPU Cover Panel; FRU 7: Battery

Conditions:
NER-0, RPE-1 : Report error
CON-0, STP-1 : Action on error
SV1-1, SV2-2 : Severity level
VOF-0, VON-1 : Virtual mode
RHP-0, RHU-1 : Halt protection
ROM-0, RAM-2, FAST-3 : Run environment

Table 4–5 (Cont.): KA660 Console Displays and FRU Pointers

No.	Test	LED	Description	FRUs	Conditions
script_A8:					
30	30	8	map_setup	2, 1	RPE-STP-SV2-VOF-RHP-ROM
49	49	8	memory_test_fdm	2, 1, 3	RPE-STP-SV2-VOF-RHP-ROM
script_A9:					
4F	4F	8	memory_data	2, 1, 3	RPE-STP-SV2-VOF-RHP-ROM
4E	4E	8	memory_byte	2, 1, 3	RPE-STP-SV2-VOF-RHP-FAST
4D	4D	8	memory_addr	2, 1, 3	RPE-STP-SV2-VOF-RHP-FAST
4C	4C	8	memory_ecc_error	2, 1, 3	RPE-STP-SV2-VOF-RHP-FAST
4B	4B	8	mask_write_w_errs	2, 1, 3	RPE-STP-SV2-VOF-RHP-FAST
47	47	8	memory_refresh	2, 1, 3	RPE-STP-SV2-VOF-RHP-FAST
40	40	8	count_bad_pages	2, 1, 3	RPE-CON-SV2-VOF-RHU-ROM
41	41	C	board_reset	2, 1, 3	RPE-CON-SV2-VOF-RHU-ROM
script_B5:					
46	46	B	SOC_cache_diag_mode	1	RPE-CON-SV2-VOF-RHP-ROM
46	46	B	SOC_cache_diag_mode	1	RPE-CON-SV2-VOF-RHP-ROM
46	46	B	SOC_cache_diag_mode	1	RPE-CON-SV2-VOF-RHP-ROM
46	46	B	SOC_cache_diag_mode	1	RPE-CON-SV2-VOF-RHP-ROM
46	46	B	SOC_cache_diag_mode	1	RPE-CON-SV2-VOF-RHP-ROM
46	46	B	SOC_cache_diag_mode	1	RPE-CON-SV2-VOF-RHP-ROM
46	46	B	SOC_cache_diag_mode	1	RPE-CON-SV2-VOF-RHP-ROM
44	44	B	SOC_D_Cache_w_memory	1, 2	RPE-CON-SV2-VOF-RHP-ROM
44	44	B	SOC_D_Cache_w_memory	1, 2	RPE-CON-SV2-VOF-RHP-ROM

Field-replaceable Units:
FRU 1: KA660; FRU 2: MS650; FRU 3: Backplane; FRU 4: Q22-Bus Device; FRU 5: System Power Supply; FRU 6: CPU Cover Panel; FRU 7: Battery

Conditions:
NER-0, RPE-1 : Report error
CON-0, STP-1 : Action on error
SV1-1, SV2-2 : Severity level
VOF-0, VON-1 : Virtual mode
RHP-0, RHU-1 : Halt protection
ROM-0, RAM-2, FAST-3 : Run environment

Table 4–5 (Cont.): KA660 Console Displays and FRU Pointers

No.	Test	LED	Description	FRUs	Conditions
script_B5:					
44	44	B	SOC_D_Cache_w_memory	1, 2	RPE-CON-SV2-VOF-RHP-ROM
44	44	B	SOC_D_Cache_w_memory	1, 2	RPE-CON-SV2-VOF-RHP-ROM
44	44	B	SOC_D_Cache_w_memory	1, 2	RPE-CON-SV2-VOF-RHP-ROM
44	44	B	SOC_D_Cache_w_memory	1, 2	RPE-CON-SV2-VOF-RHP-ROM
44	44	B	SOC_D_Cache_w_memory	1, 2	RPE-CON-SV2-VOF-RHP-ROM
44	44	B	SOC_D_Cache_w_memory	1, 2	RPE-CON-SV2-VOF-RHP-ROM
45	45	7	cache_mem_cqbic	1, 2	RPE-CON-SV2-VOF-RHP-ROM
45	45	7	cache_mem_cqbic	1, 2	RPE-CON-SV2-VOF-RHP-ROM
45	45	7	cache_mem_cqbic	1, 2	RPE-CON-SV2-VOF-RHP-ROM
45	45	7	cache_mem_cqbic	1, 2	RPE-CON-SV2-VOF-RHP-ROM
45	45	7	cache_mem_cqbic	1, 2	RPE-CON-SV2-VOF-RHP-ROM
45	45	7	cache_mem_cqbic	1, 2	RPE-CON-SV2-VOF-RHP-ROM
45	45	7	cache_mem_cqbic	1, 2	RPE-CON-SV2-VOF-RHP-ROM
45	45	7	cache_mem_cqbic	1, 2	RPE-CON-SV2-VOF-RHP-ROM
43	43	B	SOC_DI_Cache_w_memory	1, 2	RPE-CON-SV2-VOF-RHP-FAST
43	43	B	SOC_DI_Cache_w_memory	1, 2	RPE-CON-SV2-VOF-RHP-FAST
43	43	B	SOC_DI_Cache_w_memory	1, 2	RPE-CON-SV2-VOF-RHP-FAST
43	43	B	SOC_DI_Cache_w_memory	1, 2	RPE-CON-SV2-VOF-RHP-FAST
43	43	B	SOC_DI_Cache_w_memory	1, 2	RPE-CON-SV2-VOF-RHP-FAST
43	43	B	SOC_DI_Cache_w_memory	1, 2	RPE-CON-SV2-VOF-RHP-FAST
43	43	B	SOC_DI_Cache_w_memory	1, 2	RPE-CON-SV2-VOF-RHP-FAST
43	43	B	SOC_DI_Cache_w_memory	1, 2	RPE-CON-SV2-VOF-RHP-FAST
43	43	B	SOC_DI_Cache_w_memory	1, 2	RPE-CON-SV2-VOF-RHP-ROM
43	43	B	SOC_DI_Cache_w_memory	1, 2	RPE-CON-SV2-VOF-RHP-FAST
99	99	B	flush_ena_caches	1, 2	RPE-CON-SV2-VOF-RHP-ROM

Field-replaceable Units:
FRU 1: KA660; FRU 2: MS650; FRU 3: Backplane; FRU 4: Q22-Bus Device; FRU 5: System Power Supply; FRU 6: CPU Cover Panel; FRU 7: Battery

Conditions:
NER-0, RPE-1 : Report error
CON-0, STP-1 : Action on error
SV1-1, SV2-2 : Severity level
VOF-0, VON-1 : Virtual mode
RHP-0, RHU-1 : Halt protection
ROM-0, RAM-2, FAST-3 : Run environment

4.3.5 System Halt Messages

Table 4–6 lists messages that may appear on the console terminal when a system error occurs.

Table 4–6: System Halt Messages

Code	Message	Explanation
?02	EXT HLT	External halt, caused by either console BREAK condition, Q22-bus BHALT_L, or DBR<AUX_HLT> bit was set while enabled.
_03	——	Power-up, no halt message is displayed. However, the presence of the firmware banner and diagnostic countdown indicates this halt reason.
?04	ISP ERR	In attempting to push state onto the interrupt stack during an interrupt or exception, the processor discovered that the interrupt stack was mapped NO ACCESS or NOT VALID.
?05	DBL ERR	The processor attempted to report a machine check to the operating system, and a second machine check occurred.
?06	HLT INST	The processor executed a HALT instruction in kernel mode.
?07	SCB ERR3	The SCB vector had bits <1:0> equal to 3.
?08	SCB ERR2	The SCB vector had bits <1:0> equal to 2.
?0A	CHM FR ISTK	A change mode instruction was executed when PSL<IS> was set.
?0B	CHM TO ISTK	The SCB vector for a change mode had bit 0 set.
?0C	SCB RD ERR	A hard memory error occurred while the processor was trying to read an exception or interrupt vector.
?10	MCHK AV	An access violation or an invalid translation occurred during machine check exception processing.
?11	KSP AV	An access violation or translation not valid occurred during processing of a kernel stack not valid exception.
?12	DBL ERR2	Double machine check error. A machine check occured while trying to service a machine check.
?13	DBL ERR3	Double machine check error. A machine check occured while trying to service a kernel stack-not-valid exception.
?19	PSL EXC5[1]	PSL<26:24> = 5 on interrupt or exception.
?1A	PSL EXC6[1]	PSL<26:24> = 6 on interrupt of exception.
?1B	PSL EXC7[1]	PSL<26:24> = 7 on interrupt or exception.
?1D	PSL REI5[1]	PSL<26:24> = 5 on an rei instruction
?1E	PSL REI6[1]	PSL<26:24> = 6 on an rei instruction.
?1F	PSL REI7[1]	PSL<26:24> = 7 on an rei instruction.
?3F	MICROVERIFY FAILURE	Microcode power-up self-test failed.

4.3.6 Console Error Messages

Table 4–7 lists messages issued in response to an error or to a console command that was entered incorrectly.

Table 4–7: Console Error Messages

Code	Message	Description
?61	CORRUPTION	The console program database has been corrupted.
?62	ILLEGAL REFERENCE	Illegal reference. The requested reference would violate virtual memory protection, the address is not mapped, the reference is invalid in the specified address space, or the value is invalid in the specified destination.
?63	ILLEGAL COMMAND	The command string cannot be parsed.
?64	INVALID DIGIT	A number has an invalid digit.
?65	LINE TOO LONG	The command was too large for the console to buffer. The message is issued only after receipt of the terminating carriage return.
?66	ILLEGAL ADRRESS	The address specified falls outside the limits of the address space.
?67	VALUE TOO LARGE	The value specified does not fit in the destination.
?68	QUALIFIER CONFLICT	Qualifier conflict, for example, two different data sizes are specified for an EXAMINE command.
?69	UNKNOWN QUALIFIER	The switch is unrecognized.
?6A	UNKNOWN SYMBOL	The symbolic address in an EXAMINE or DEPOSIT command is unrecognized.
?6B	CHECKSUM	The command or data checksum of an X command is incorrect. If the data checksum is incorrect, this message is issued, and is not abbreviated to "Illegal command".
?6C	HALTED	The operator entered a HALT command.
?6D	FIND ERROR	A FIND command failed either to find the RPB or 128 kb of good memory.
?6E	TIME OUT	During an X command, data failed to arrive in the time expected (60 seconds).
?6F	MEMORY ERROR	A machine check occurred with a code indicating a read or write memory error.
?70	UNIMPLEMENTED	Unimplemented function.
?71	NO VALUE QUALIFIER	Qualifier does not take a value.
?72	AMBIGUOUS QUALIFIER	There were not enough unique characters to determine the qualifier.
?73	VALUE QUALIFIER	Qualifier requires a value.
?74	TOO MANY QUALIFIERS	Too many qualifiers supplied for this command.
?75	TOO MANY ARGUMENTS	Too many arguments supplied for this command.
?76	AMBIGUOUS COMMAND	There were not enough unique characters to determine the command.

Table 4–7 (Cont.): Console Error Messages

Code	Message	Description
?77	TOO FEW ARGUMENTS	Insufficient arguments supplied for this command.
?78	TYPEAHEAD OVERFLOW	The typeahead buffer overflowed.
?79	FRAMING ERROR	A framing error was detected on the console serial line.
?7A	OVERRUN ERROR	An overrun error was detected on the console serial line.
?7B	SOFT ERROR	A soft error occurred.
?7C	HARD ERROR	A hard error occurred.
?7D	MACHINE CHECK	A machine check occurred.

4.3.7 VMB Error Messages

Table 4–8 lists the boot error messages and their descriptions.

Table 4–8: VMB Error Messages

Message Number	Mnemonic	Interpretation
?40	NOSUCHDEV	No bootable devices found.
?41	DEVASSIGN	Device is not present.
?42	NOSUCHFILE	Program image not found.
?43	FILESTRUCT	Invalid boot device file structure.
?44	BADCHKSUM	Bad checksum on header file.
?45	BADFILEHDR	Bad file header.
?46	BADIRECTORY	Bad directory file.
?47	FILNOTCNTG	Invalid program image format.
?48	ENDOFFILE	Premature end of file encountered.
?49	BADFILENAME	Bad file name given.
?4A	BUFFEROVF	Program image does not fit in available memory.
?4B	CTRLERR	Boot device I/O error.
?4C	DEVINACT	Failed to initialize boot device.
?4D	DEVOFFLINE	Device is offline.
?4E	MEMERR	Memory initialization error.
?4F	SCBINT	Unexpected SCB exception or machine check.
?50	SCB2NDINT	Unexpected exception after starting program image.
?51	NOROM	No valid ROM image found.
?52	NOSUCHNODE	No response from load server.
?53	INSFMAPREG	Invalid memory configuration.
?54	RETRY	No devices bootable, retrying.
?55	IVDEVNAM	Invalid device name.

Table 4–8 (Cont.): VMB Error Messages

Message Number	Mnemonic	Interpretation
?56	DRVERR	Drive error.

4.4 Acceptance Testing

Perform the acceptance testing procedure listed below, after installing a system or whenever replacing the following:

KA660 module
MS650 module
Memory data interconnect cable
Backplane
DSSI drive
H3602–00

1. Run five error-free passes of the power-up scripts by entering the following command:

   ```
   >>> R T 0
   ```

   Press CTRL/C to terminate the scripts.

2. Perform the next two steps for a granular test of memory.

   ```
   >>> T A8
   >>> R T A7
   ```

   The first command runs script A8 for one pass. This command enables mapping out of solid single-bit ECC as well as multi-bit ECC errors. It will also run script A7 for one pass.

   The second command runs script A7 repeatedly. This command runs the memory tests only and does not reset the bitmap. Press CTRL/C after two passes to terminate the script. This test takes up to 5 minutes per pass, depending on the amount of memory in the system. Most of the memory diagnostics test memory on a page boundary.

   If any of the memory tests fail, they mark the bitmap and continue with no error printout to the console. An exception is test 40 (count bad pages). If any single-bit or multi-bit ECC errors are detected, they are reported in test 40. Such a failure indicates that pages in memory have been marked bad in the bitmap because of solid single-bit and/or multi-bit ECC errors.

3. Check the memory configuration again, since test 31 can check for only a few invalid configurations. For example, test 31 cannot report that a memory board is missing from the configuration, since it has no way of knowing if the board should be there or not. In the following four examples, the SHOW MEMORY/FULL command is shown first without errors and then with errors inserted.

```
>>>SHOW MEMORY/FULL

Memory 0: 00000000 to 00FFFFFF, 16MB, 0 bad pages

Total of 16MB, 0 bad pages, 104 reserved pages

Memory Bitmap
-00FF3000 to 00FF3FFF, 8 pages

Console Scratch Area
-00FF4000 to 00FF7FFF, 32 pages

Qbus Map
-00FF8000 to 00FFFFFF, 64 pages

Scan of Bad Pages
```

The following command deposits errors into memory. `>>>D FF3100 9/9/0` Running SHOW MEMORY/FULL this time points out the errors.

```
>>>SHOW MEMORY/FULL

Memory 0: 00000000 to 00FFFFFF, 16MB, 32 bad pages

Total of 16MB, 32 bad pages, 104 reserved pages

Memory Bitmap
-00FF3000 to 00FF3FFF, 8 pages

Console Scratch Area
-00FF4000 to 00FF7FFF, 32 pages

Qbus Map
-00FF8000 to 00FFFFFF, 64 pages

Scan of Bad Pages
-00100000 to 00103FFF, 32 pages
>>>
```

Memories 0 through 3 are the MS650 memory modules. The Q22-bus map always spans the top 32 Kbytes of good memory. The memory bitmap always spans two pages (1 Kbyte) per 4 Mbytes of memory configured.

Use utility 9C to compare the contents of configuration registers MEMCSR 0–15 with the memory installed in the system:

```
>>>T 9C
  SBR=017B8000    SLR=00002021  SAVPC=20044827 SAVPSL=04190304    SCBB=20052400
 P0BR=80000000   P0LR=00100A80   P1BR=0A0A0A08   P1LR=000B0B0B     SID=14000006
 TODR=0010E085   ICCS=00000000                  MAPEN=00000000   BDMTR=20084000
 TCR0=00000000   TIR0=00000000  TNIR0=00000000  TIVR0=00000078   BDMKR=0000007C
 TCR1=00000001   TIR1=0052680A  TNIR1=0000000F  TIVR1=0000007C
 RXCS=00000000   RXDB=0000000D   TXCS=00000000   TXDB=00000030
  SCR=0000D000   DSER=00000000  QBEAR=0000000F   DEAR=00000000   QBMBR=017F8000
  BDR=08D0EFFF  DLEDR=0000000C  SSCCR=00D55537  CBTCR=00000004   IPCR0=0000
DSSI_0=00  (BUS_0)     PQBBR_0=03060022    PMCSR_0=00000000    SSHMA_0=0000CA20
     PSR_0=00000000     PESR_0=00000000     PFAR_0=00000000     PPR_0=00000000
NICSR0=1FFF0003   3=00004030   4=00004050   5=8039FF00   6=83E0F000  7=00000000
NICSR9=04E204E2  10=00040000  11=00000000  12=00000000  13=00000000 15=0000FFFF
NISA=08-00-2B-12-BC-AC     RDES0=00441300   1=00000000   2=05EE0000  3=000046F0
                           TDES0=00008C80   1=07000000   2=00400000  3=000040FA
MEM_FRU 1      MCSR_0=80000017      1=80400017      2=80800017       3=80C00017
MEM_FRU 2      MCSR_4=81000016      5=81400016      6=00000016       7=00000016
MEM_FRU 3      MCSR_8=00000000      9=00000000     10=00000000      11=00000000
MEM_FRU 4      MCSR12=00000000     13=00000000     14=00000000      15=00000000
MEMCSR17=00000013  MEMCSR16=00000044  CSR16_page_address=00000000
MSER=00000000  CCR=00000014
```

One memory bank is enabled for each 4 Mbytes of memory. The MEMCSRs map modules as follows:

MEMCSR 0-3	First MS650 memory module
MEMCSR 4-7	Second MS650 memory module
MEMCSR 8-11	Third MS650 memory module
MEMCSR 12-15	Fourth MS650 memory module

Verify the following:

- If a memory board is not present, bits <31:0> are all zeros for the corresponding group of four MEMCSRs. See the values for MEMCSR 8–11 in the example.
- Bits <25:22> should increment by one starting at zero in any group of four MEMCSRs whose bit 31 equals 1. In the example above, bits <25:22> of MEMCSR 4 and 5 increment by one, resulting in an increment of four in their longwords. If bit 31 equals 0, <25:22> should equal zero.

4. Check the Q22-bus and the Q22-bus logic in the KA660 CQBIC chip, and the configuration of the Q22-bus, as follows:

```
>>> SHOW QBUS
Scan of Qbus I/O Space
-20000120 (760440) = 0080 DHQ11/DHV11/CXA16/CXB16/CXY08
-20000122 (760442) = F081
-20000124 (760444) = DD18
-20000126 (760446) = 0200
-20000128 (760450) = 0000
-2000012A (760452) = 0000
-2000012C (760454) = 8000
-2000012E (760456) = 0000
-20001920 (774440) = FF08 DESQA
-20001922 (774442) = FF00
-20001924 (774444) = FF2B
-20001926 (774446) = FF06
-20001928 (774450) = FF16
-2000192A (774452) = FFF2
-2000192C (774454) = 00F8
-2000192E (774456) = 1030
-20001940 (774500) = 0000 TQK50/TQK70/TU81E/RV20/K-TAPE
-20001942 (774502) = 0BC0
-20001F40 (777500) = 0020 IPCR

Scan of Qbus Memory Space
>>>
```

The columns are described below. The examples listed are from the last line of the example above.

First column = the VAX I/O address of the CSR, in hexadecimal (20001F40).
Second column = the Q22-bus address of the CSR, in octal (777500).
Third column = the data, contained at the CSR address, in hexadecimal (0020).
Fourth column = the device vector in octal, according to the fixed or floating Q22-bus and UNIBUS algorithm (004).
Fifth column = the device name (IPCR, the KA660 interprocessor communications register).

Additional lines for the device are displayed if more than one CSR exists.

The last line, Scan of Qbus Memory Space, displays memory residing on the Q22-bus, if present. VAX memory mapped by the Q-22 bus map is not displayed.

If the system contains an MSCP or TMSCP controller, run test 81. This test performs step one of the UQ port initialization sequence, performs the SA wraparound test, and checks the Q-22 bus interrupt logic.

NOTE: *This test will erroneously generate messages indicating the KFQSA module has failed.*

If you do not specify the CSR address, the test searches for and runs on the first MSCP device by default. To test the first TMSCP device, you must specify the first parameter:

```
>>> T 81 20001940
```

You can specify other addresses if you have multiple MSCP or TMSCP devices in the first parameter. This action may be useful to isolate a problem with a controller, the KA660, or the backplane. Use the VAX address provided by the SHOW QBUS command to determine the CSR value. If you do not specify a value, the MSCP device at address 20001468 is tested by default.

5. Check that all UQSSP, MSCP, TMSCP, and Ethernet controllers and devices are visible by entering the following command:

```
>>> SHOW DEVICE

DSSI Bus 0 Node 0 (R3YRME)
-DIA0 (RF31)

DSSI Bus 0 Node 1 (R3VBNC)
-DIA1 (RF31)

DSSI Bus 0 Node 7 (*)

UQSSP Tape Controller 0 (774500)
-MUA0 (TK70)

Ethernet Adapter
-EZA0 (08-00-2B-08-E8-6E)

Ethernet Adapter 0 (774440)
-XQA0 (08-00-2B-06-16-F2)
```

In the above example, the console displays the remote DSSI node names and node numbers of two ISE controllers it recognizes. The lines below each node name and number are the logical unit numbers of any attached devices, DIA0 and DIA1 in this case.

DSSI Node 7 (*) is the KA660 DSSI adapter. In most cases, the KA660 is the local DSSI node shown by the asterisk and has a node number of 7. DSSI node names, node numbers, and unit numbers should be unique.

The UQSSP (TQK70) tape controller and its CSR address are also shown. The line below this display shows a TK70 connected.

The next two lines show the logical name and station address for the KA660 Ethernet adapter.

The last two lines refer to DESQA controller, the Q22-bus CSR address, logical name (XQA0), and the station address.

6. Test the DSSI subsystem using the KA660 ROM-based Diagnostics and Utilities Protocol (DUP) facility. This facility allows you to connect to the DUP server in the RF drive controller. Examples follow.

```
>>> SET HOST /DUP/DSSI 7
Starting DUP server...

Stopping DUP server...
```

In this example, a DUP connection was made with DSSI node 7, the KA660. The DUP server times out, since no local programs exist and no response packet was received.

```
>>> SET HOST /DUP/DSSI 1
Starting DUP server...

DSSI Bus 0 Node 1 (R3VBNC)
DRVEXR V1.0  D 21-FEB-1988 21:27:54
DRVTST V1.0  D 21-FEB-1988 21:27:54
HISTRY V1.0  D 21-FEB-1988 21:27:54
ERASE  V1.0  D 21-FEB-1988 21:27:54
PARAMS V1.0  D 21-FEB-1988 21:27:54
DIRECT V1.0  D 21-FEB-1988 21:27:54
End of directory

Task Name? DRVTST
Write/read anywhere on medium? [1=Yes/(0=No)]: <CR>
5 minutes for test to complete.
Compare failed on head  1 track 1091.
Compare failed on head  0 track  529.

Task Name? DRVEXR
Write/read anywhere on medium? [1=Yes/(0=No)]: <CR>
Test time in minutes? [(10)-100]:
10 minutes for test to complete.
R3VBNC::MSCP$DUP 21-FEB-1988 21:37:35 DRVEXR CPU=00:00:01.88 PI=43
R3VBNC::MSCP$DUP 21-FEB-1988 21:37:38 DRVEXR CPU=00:00:03.38 PI=79
Compare failed on head  1 track 1091.
R3VBNC::MSCP$DUP 21-FEB-1988 21:37:40 DRVEXR CPU=00:00:04.97 PI=116
^C
>>>
```

In the above example, the local programs DRVTST and DRVEXR are run on drive 1.

CAUTION: *Do not enter 1 in response to the question* `Write/read anywhere on medium?`. *Doing so will destroy data on the disk.*

Press Return, which uses the default, allowing reads and writes to the DBNs only. CTRL/T or CTRL/G displays a message as shown in the DRVEXR example above (the lines beginning with R3VBNC::). In the example, CTRL/T has been pressed twice, to show the difference in the time and in the value of the progress indicator (PI).

Press CTRL/C to terminate the program.

Use the local programs PARAMS (Section 4.8.5) and HISTRY (Section 4.8.3) to determine the cause of errors displayed during DRVTST or DRVEXR. DRVTST should run successfully for one pass on each drive.

7. If there are one or more DELQA modules in the system, use test 82 to invoke the Ethernet option's self-test and receive status from the host firmware. Test 82 is useful for acceptance testing if you cannot access the system enclosure to see the DELQA LEDs.

8. After the above steps have completed successfully, load MDM and run the system tests from the Main Menu. If they run successfully, the system has gone through its basic checkout and you can load the software.

4.5 Troubleshooting

This section contains suggestions for determining the cause of ROM-based diagnostic test failures.

4.5.1 FE Utility

The FE utility dumps the diagnostic state to the console as shown below.

```
>>> T FE

 Bitmap=00FF3000, Length=00001000, Checksum=807F, Busmap=00FF8000
 Test_number=41, Subtest=00, Loop_Subtest=00, Error_type=00
 Error_vector=0000, Last_exception_PC=00000000, Severity=02
 Total_error_count=0000, Led_display=0C, Console_display=03, save_mchk_code=80
 parameter_1=00000000  2=00000000  3=00000000  4=00000000  5=00000000
 parameter_6=00000000  7=00000000  8=00000000  9=00000000 10=00000000
 previous_error=00000000, 00000000, 00000000, 00000000
 Flags=00FFFC10440E, SET_mask=FF
 Return_stack=201406D4, Subtest_pc=20062730, Timeout=00030D40
>>>
```

The most useful fields displayed above are as follows:

- Error_type_vector. The SCB vector through which the unexpected interrupt or exception trapped if error_type equals FE, FD, FC, or EF.
- Total_error_count. Four hexadecimal digits showing the number of previous errors that have occurred.

- Previous_error. Contains the history of the last four errors. Each longword contains four bytes of information. From left to right these are the error_type, subtest_log, test number, and loop_subtestlog.
- Save machine check code (save_mchk_code). Valid only if the test halts on error. This field has the same format as the hardware error summary register.
- Parameters 1 through 10. Valid on the last text run.
- Last_exception_PC. PC of exception if error_type is FE, FD, FC, or EF.

4.5.2 Isolating Memory Failures

This section describes procedures for isolating memory subsystem failures, particularly when the system contains more than one MS650 memory module.

1. SHOW MEMORY/FULL

   Use the SHOW MEMORY/FULL command to examine failures detected by the memory tests. Use this command if test 40 fails, which indicates that pages have been marked bad in the bitmap. SHOW MEMORY/FULL will break out the number of pages marked bad if any on each individual memory board present.

2. T A9

   Script A9 runs only the memory tests and halts on the first hard or soft error found. This script will not continue after an error, and it will not fully mark the bitmap with all errors. The main purpose for A9 is to find the first test that caused an error and print out the error message. The user can then determine the specific error with a listing for the test. Script A9 is generally not needed to determine a failing field- replaceable unit (FRU).

3. Continue on Error flag.

   The fourth parameter allows the user to determine, after a memory error, if a test should continue or stop immediately to allow a printout. All scripts except A9 have Continue on Error set to allow the memory bitmap to be fully marked correctly for all errors. When running an individual memory test, the default for parameter 4 is to stop on any error. You have to specifically set parameter 4 to a 1 to enable Continue on Error.

   When Continue on Error is 0, there is no retry after a soft or a hard error. In other words, if you run any memory test (47, 48, 4A, or 4F) with the default value for parameter 4 (0), or if you run the A9 script,

any error, hard or soft, will cause the test to stop, print an error, and also mark the bitmap. The bitmap is always marked in 256 KB sections to allow tests to run quicker when errors occur.

The memory tests have a soft error counter for each board. These counters are only incremented if a soft error occurs during a test and Continue on Error is enabled (1). The soft error counters are also not incremented during the A6 script, which only marks multiple bit errors. The A6 script should not normally be used because it will not mark off hard single-bit errors.

Soft errors are defined in these tests as single-bit errors, and when the data is rewritten, no error occurs on the next read of the data.

4. Running an individual test

   Parameters 1 and 2 determine the starting address for each memory test. Use the SHOW MEMORY command to print out the addresses for all boards. The first board is board 0.

   Instead of an address, you can also enter a starting and ending board number. The first board is number 1. After the test starts, board numbers are replaced with the actual addresses.

   You can also change the address increment parameter 3 for each memory tests. Tests 4F, 4E, 4A, 4B and 4C run very slowly. The normal address increment for these tests would 256 Kbytes (0x40000) or greater. Smaller increments would normally be used when selecting a smaller address range from starting to ending address. For example, run test 4F on the second memory board, increment address by 100000 hexadecimal (1 MByte), and continue testing if an error occurs.

```
>>>T 4F 2 2 100000 1
```

   Run test 4C on every location from address 0x40000 to 0x4FFFF. Stop on the first error if any. End address actually specifies the last byte location + 1 to test.

```
>>>T 4C 40000 50000 8
```

5. T 40

   The SHOW MEMORY command displays pages that are marked bad by the memory tests and is easier to interpret than test 40. There is only one instance in which test 40 reports information that SHOW MEMORY does not report. Test 40 reports the number of soft errors that have been counted by the memory tests, if any, for each memory board. The default when running test 40 is to ignore soft errors. To count soft errors, enter the following command:

```
>>>T 40 1 4 0
```

This command causes all soft and hard errors to be checked against all memory boards present. For soft errors the limit to check against is 0, which is the third parameter. If test 40 fails with SUBTESTLOG = 07, then R5-R8 in the error dump list refers to soft errors for boards 1 through 4.

6. T 9C

   The utility 9C is useful after system crashes or similar events because it dumps the current contents of most CPU registers on the KA660.

To help in isolating an FRU, examine registers MEMCSR 0–15 by entering T 9C at the console I/O mode prompt (Example 4–5). Utility 9C is also useful for examining the error registers MSER, CACR, DSER, and MEMCSR16, upon a fatal system crash or similar event. See Example 4–5 for an example of T 9C.

Example 4–5: T 9C

```
>>>>>>T 9C
  SBR=017B8000    SLR=00002021  SAVPC=20044827 SAVPSL=04190304    SCBB=20052400
 P0BR=80000000   P0LR=00100A80   P1BR=0A0A0A08   P1LR=000B0B0B     SID=14000006
 TODR=0010E085   ICCS=00000000                  MAPEN=00000000   BDMTR=20084000
 TCR0=00000000   TIR0=00000000  TNIR0=00000000  TIVR0=00000078   BDMKR=0000007C
 TCR1=00000001   TIR1=0052680A  TNIR1=0000000F  TIVR1=0000007C
 RXCS=00000000   RXDB=0000000D   TXCS=00000000   TXDB=00000030
  SCR=0000D000   DSER=00000000  QBEAR=0000000F   DEAR=00000000   QBMBR=017F8000
  BDR=08D0EFFF  DLEDR=0000000C  SSCCR=00D55537  CBTCR=00000004   IPCR0=0000
DSSI_0=00  (BUS_0)     PQBBR_0=03060022    PMCSR_0=00000000    SSHMA_0=0000CA20
     PSR_0=00000000     PESR_0=00000000     PFAR_0=00000000     PPR_0=00000000
NICSR0=1FFF0003   3=00004030   4=00004050   5=8039FF00   6=83E0F000   7=00000000
NICSR9=04E204E2  10=00040000  11=00000000  12=00000000  13=00000000 15=0000FFFF
NISA=08-00-2B-12-BC-AC     RDES0=00441300   1=00000000   2=05EE0000   3=000046F0
                           TDES0=00008C80   1=07000000   2=00400000   3=000040FA
MEM_FRU 1      MCSR_0=80000017      1=80400017      2=80800017      3=80C00017
MEM_FRU 2      MCSR_4=81000016      5=81400016      6=00000016      7=00000016
MEM_FRU 3      MCSR_8=00000000      9=00000000     10=00000000     11=00000000
MEM_FRU 4      MCSR12=00000000     13=00000000     14=00000000     15=00000000
MEMCSR17=00000013  MEMCSR16=00000044  CSR16_page_address=00000000
MSER=00000000  CCR=00000014

>>>
                           3      2   2
                           1      5   2
MEMCSR16 = 8094000F hex = 1000 0000 1001 0100 0000 0000 0000 1111
                                  || ||
MEMCSR_5 = 80800016 hex = 1000 0000 1000 0000 0000 0000 0001 0110
                          ^         ^
                bit 31 set        25:22 match
```

4.5.3 Additional Troubleshooting Suggestions

Note the following additional suggestions when diagnosing a possible memory failure.

- If more than one memory module is failing, you should suspect the KA660 module, CPU/memory cable, backplane, or MS650 modules as the cause of failure.
- Always check the seating of the memory cable first before replacing a KA660 or MS650 module. If the seating appears to be improper, rerun the tests. Also remember to leave the middle connector disconnected for a three-connector cable when the system is configured with only one MS650.
- If you are rotating MS650 modules to verify that a particular memory module is causing the failure, be aware that a module may fail in a different way when in a different slot.
- Be sure to put the modules back in their original positions when you are finished.
- If memory errors are found in the error log, use the KA660 ROM-based diagnostics to see if it is an MS650 problem, or if it is related to the KA660, CPU/memory interconnect cable, or backplane. Follow steps 1–3 of Section 4.4 and Section 4.5.2 to aid in isolating the failure.
- Use the SHOW QBUS, SHOW DEVICE, and SET HOST/DUP commands when troubleshooting I/O subsystem problems.
- Use the CONFIG command to help with configuration problems or when installing new options onto the Q-bus. See the command descriptions in Chapter 3.
- You can run a DSSI device power-up diagnostic without performing a cold restart or spinning the disk drives down and back up.

  Enter the following at the console prompt: `>>>T 58 {node_number}`

  A CI Reset command is issued to the DSSI device, causing the device to perform its power-up diagnostics.

  Parameter 1 is the DSSI node ID or port number. It must be in the range of 0–7 (0 is the default). Use the default for parameter 2.

  You can perform this test repeatedly with the REPEAT command (`R T 58 {node_ID}`). In that case the drive's self-tests run repeatedly until you press CTRL/C to terminate the test.

- Once the test has completed successfully, you can examine the DSSI device's internal error logs by running the DUP local programs HISTRY and PARAMS. Refer to Section 4.8.3 and Section 4.8.5 for further information.

4.6 Loopback Tests and Fuse Problems

You can use external loopback tests to localize problems with the Ethernet, console, and DSSI subsystems.

Check that dc power and pico fuses on the KA660 are functioning correctly. Three 1.5-A pico fuses (PN 12–10929–08) are located near the handle on the component side of the KA660 module, as shown in Figure 1–1. The fuses are numbered from left to right as follows:

F1, F2: Backplane fingers
F3: Memory and I/O connectors

Replace the fuse, not the KA660, if a fuse has gone bad. Table 4–9 lists some symptoms of faulty fuses.

Table 4–9: KA660 Fuses

Bad Fuse	Symptom
F1 bad (+5 V)	Cover panel hexadecimal LED display is off.
F2 bad (+12 V)	Both Thinwire and standard Ethernet LEDs on the CPU cover panel are off.
F3 bad (DSSI Term)	DSSI terminator LED is off

Ethernet external loopback test 5F fails (for ThinWire only, since the fuse protects +12 V supplied to the DESTA on the CPU cover panel).

The LED on the loopback connector (PN 12–22196–02) for standard Ethernet is off; external loopback tests for standard Ethernet pass, however.

Console SLU external loopback test fails.

Only local DSSI node (typically node 7 for the KA660) is reported by SHOW DEVICE or SHOW DSSI commands.

DSSI external loopback test 56 fails.

DSSI Problems

For DSSI problems, run the SHAC external loopback test (test 56). To check the DSSI bus out to the KA660 connector, plug one end of the test cable (PN 17–02216–01) to the H3281 loopback connector and the other end to the KA660 DSSI connector. To test out to the end of the DSSI bus, turn off the system, remove all DSSI devices with the exception of the KA660

from the bus, and plug the external DSSI loopback connector in place of the DSSI bus terminator.

Ethernet Problems

For ThinWire Ethernet problems, run the external loopback test (NI test, number 5F) by entering the following:

```
>>> T 5F 1
```

Set parameter 1 to run this test. Only the external loopback test runs. Be sure to set the Ethernet Connector switch on the CPU cover panel to the ThinWire position. Use two 50-ohm H8225 terminators connected to an H8223 T-connector.

To test the standard Ethernet connector, use loopback connector (PN 12–22196–02) in conjunction with MDM.

4.6.1 Testing the Console Port

To test the console port at power-up, set the Power-Up Mode switch on the CPU cover panel to the Test position, and install an H3103 loopback connector into the MMP of the cover panel. The H3103 connects the console port transmit and receive lines. At power-up, the SLU_EXT_LOOPBACK IPT then runs a continuous loopback test.

While the test is running, the LED display on the CPU I/O insert should alternate between 6 and 3. A value of 6 in the display indicates a test failure. If the test fails, one of the following parts is faulty: the KA660 CPU module, the CPU cover panel, or the cabling.

To test out to the end of the console terminal cable:

1. Plug the MMJ end of the console terminal cable into the CPU cover panel.
2. Disconnect the other end of the cable from the terminal.
3. Place an H8572 adapter into the disconnected end of the cable.
4. Connect the H3103 to the H8572.

4.7 Module Self-Tests

Module self-tests run when you turn on the system. A module self-test can detect hard or repeatable errors, but usually not intermittent errors.

Module LEDs display pass/fail test results.

A pass by a module self-test does not guarantee that the module is good, because the test usually checks only the controller logic. The test usually does not check the module Q22-bus interface, the line drivers and receivers, or the connector pins—all of which have relatively high failure rates.

A fail by a module self-test is accurate, because the test does not require any other part of the system to be working.

The following modules do not have LED self-test indicators:

DFA01
DPV11
DRQ3B
DZQ11
KLESI
LPV11
TSV05

The following modules have one green LED, which indicates that the module is receiving +5 and +12 Vdc:

CXA16
CXB16
CXY08
KZQSA

Table 4–10 lists loopback connectors for common KA660 system modules.

Table 4–10: Loopback Connectors for Q22-Bus Devices

Device	Module Loopback	Cable Loopback
CXA16/CXB16	H3103 + H8572[1]	
CXY08	H3046 (50-pin)	H3197 (25-pin)
DELQA	PN 12–22196–02	
DPV11	H3259	H3260
DSSI[2]	–	–
DZQ11	PN 12–15336–00 or H325	H329 (PN 12–27351–01)
Ethernet[3]	–	–
LPV11	None	None
KA660/H3602–00	H3103	H3103 + H8572
KMV1A	H3255	H3251

[1]Use the appropriate cable to connect transmit-to-receive lines. H3101 and H3103 are double-ended cable connectors.

[2]For DSSI to KA660 or RF-series connector, use PN 17–02216–01 plus H3281 loopback. For connection to end of bus, use the DSSI loopback connector PN 12–30702–01.

[3]For ThinWire, use H8223–00 plus two H8225–00 terminators. For standard Ethernet, use PN 12–22196–02.

Table 4–10 (Cont.): Loopback Connectors for Q22-Bus Devices

Device	Module Loopback	Cable Loopback
KZQSA	PN 12-30552-02	

4.8 ISE Troubleshooting and Diagnostics

An ISE may fail either during initial power-up or during normal operation. In both cases the failure is indicated by the lighting of the red fault LED on the SCP on the enclosure front panel. The ISE also has a red fault LED, but it is not visible from the outside of the system enclosure.

If the ISE is unable to execute the Power-On Self-Test (POST) successfully, the red fault LED remains lit and the Ready indicator does not light, or both LEDs remain on.

POST is also used to handle two types of error conditions in the ISE:

1. *Controller errors* are caused by the hardware associated with the controller function of the ISE module. A controller error is fatal to the operation of the ISE, since the controller cannot establish a logical connection to the host. The red Fault indicator lights. If this occurs, replace the ISE module.

2. *Drive errors* are caused by the hardware associated with the ISE control function of the ISE module. These errors are not fatal to the ISE, since the ISE can establish a logical connection and report the error to the host. Both LEDs go out for about 1 second, then the red Fault indicator lights. In this case, run either DRVTST, DRVEXR, or PARAMS (described in the next sections) to determine the error code.

Here are three common configuration errors:

- More than one node with the same node number
- Identical bus node ID
- Identical unit numbers

The first error cannot be detected by software. Use the SHOW DSSI command to display the second and third errors. This command lists each device connected to the DSSI bus by node name and unit number.

Install the bus node ID plug in the socket on the ISE. If the ISE has no bus node ID plug, the ISE reads its bus node ID from the three-position DIP switch on the side of the ISE.

The ISE contains the following local programs (described in the following sections):

DIRECT	A directory, in DUP specified format, of available local programs
DRVTST	A comprehensive ISE functionality verification test
DRVEXR	A utility that exercises the ISE
HISTRY	A utility that saves information retained by the ISE
ERASE	A utility that erases all user data from the disk
PARAMS	A utility that allows you to look at or change ISE status, history, and parameters

A description of each local program follows, including a table showing the dialog of each program. The table also indicates the type of messages contained in the dialog, although the screen display will not indicate the message type. Message types are abbreviated as follows:

Q—Question
I—Information
T—Termination
FE—Fatal error

Access these local programs using the console SET HOST/DUP command, which creates a virtual terminal connection to the storage device and the designated local program using the Diagnostic and Utilities Protocol (DUP) standard dialog.

Once the connection is established, the local program is in control. When the program terminates, control is returned to the KA660 console. To abort or prematurely terminate a program and return control to the KA660 console, press CTRL/C or CTRL/Y.

4.8.1 DRVTST

DRVTST is a comprehensive functionality test. Errors detected by this test are isolated to the FRU level. The messages are listed in Table 4–11.

Table 4–11: DRVTST Messages

Message Type	Message
I	Copyright © 1990 Digital Equipment Corporation
Q	Write/read anywhere on the medium? [1=yes/(0=no)]
Q	User data will be corrupted. Proceed? [1=yes/(0=no)]
I	5 minutes to complete.
T	Test passed.
or	
FE	Unit is currently in use.[1]
FE	Operation aborted by user.
FE	xxxx—Unit diagnostics failed.[2]
FE	xxxx—Unit read/write test failed.[2]

[1]Either the ISE is inoperative, in use by a host, or is currently running another local program.
[2]Refer to the diagnostic error list at the end of this chapter.

Answering No to the first question ("Write/read...?") results in a read-only test. DRVTST, however, writes to a diagnostic area on the disk. Answering Yes to the first question causes the second question to be displayed.

Answering No to the second question ("Proceed?") is the same as answering No to the first question. Answering Yes to the second question permits write and read operations anywhere on the medium.

DRVTST resets the ECC error counters, then calls the timed I/O routine. After the timed I/O routine ends (5 minutes), DRVTST saves the counters again. It computes the uncorrectable error rate and byte (symbol) error rate. If either rate is too high, the test fails and the appropriate error code is displayed.

4.8.2 DRVEXR

The DRVEXR local program exercises the ISE. The test is data transfer intensive, and indicates the overall integrity of the device. Table 4–12 lists the DRVEXR messages.

Table 4–12: DRVEXR Messages

Message Type	Message
I	Copyright © 1990 Digital Equipment Corporation
Q	Write/read anywhere on the medium? [1=yes/(0=no)]
Q	User data will be corrupted. Proceed? [1=yes/(0=no)]
Q	Test time in minutes? [(10)-100]
I	ddd minutes to complete.
I	ddddddddd blocks (512 bytes) transferred.
I	ddddddddd bytes in error (soft).
I	ddddddddd uncorrectable ECC errors (recoverable).
T	Complete.
Or:	
FE	Unit is currently in use.[1]
FE	Operation aborted by user.
FE	xxxx—Unit diagnostics failed.[2]
FE	xxxx—Unit read/write test failed.[2]

[1]Either the ISE is inoperative, in use by a host, or is currently running another local program.
[2]Refer to the diagnostic error list at the end of this chapter.

Answering No to the first question (`Write/read...?`) results in a read-only test. DRVEXR, however, writes to a diagnostic area on the disk. Answering Yes to the first question results in the second question being asked.

Answering No to the second question (`Proceed?`) is the same as answering No to the first question. Answering Yes to the second question permits write and read operations anywhere on the medium.

NOTE: *If the Write-Protect switch on the SCP is pressed in (LED on) and you answer Yes to the second question, the ISE does not allow the test to run. DRVEXR displays the error message* `2006---Unit read/write test failed`. *In this case, the test has not failed, but has been prevented from running.*

DRVEXR saves the error counters, then calls the timed I/O routine. After the timed I/O routine ends, DRVEXR saves the counters again. It then reports the total number of blocks transferred, bits in error, bytes in error, and uncorrectable errors.

DRVEXR uses the same timed I/O routine as DRVTST, with two exceptions. First, DRVTST always uses a fixed time of five minutes, while you specify

the time of DRVEXR routine. Second, DRVTST determines whether the ISE is good or bad. DRVEXR reports the data but does not determine the condition of the ISE.

4.8.3 HISTRY

The HISTRY local program displays information about the history of the ISE. Table 4–13 lists the HISTRY messages.

Table 4–13: HISTRY Messages

Message Type	Field Length	Field Meaning
I	47 ASCII characters	Copyright notice
I	4 ASCII characters	Product name
I	12 ASCII characters	Drive serial number
I	6 ASCII characters	Node name
I	1 ASCII character	Allocation class
I	8 ASCII characters	Firmware revision level
I	17 ASCII characters	Hardware revision level
I	6 ASCII characters	Power-on hours
I	5 ASCII characters	Power cycles
I[1]	4 ASCII characters	Hexadecimal fault code
T		Complete

[1]Displays the last 11 fault codes as informational messages. Refer to the diagnostic error list at the end of this chapter.

The following example shows a typical screen display when you run HISTRY:

```
Copyright © 1988 Digital Equipment Corporation
RF31
EN01082
SUSAN
0
RFX V101
RF31 PCB-5/ECO-00
617
21
A04F
A04F
A103
A04F
A404
A04F
A404
```

```
A04F
A404
A04F
A404
Complete.
```

If no errors have been logged, no hexadecimal fault codes are displayed.

4.8.4 ERASE

The ERASE local program overwrites application data on the ISE while leaving the replacement control table (RCT) intact. This local program is used if an HDA must be replaced, and there is a need to protect confidential or sensitive data.

Use ERASE only if the HDA must be replaced and only after you have backed up all data.

Table 4–14 lists the ERASE messages.

Table 4–14: ERASE Messages

Message Type	Message
I	Copyright © 1988 Digital Equipment Corporation
Q	Write/read anywhere on the medium? [1=yes/(0=no)]
Q	User data will be corrupted. Proceed? [1=yes/(0=no)]
I	6 minutes to complete.
T	Complete.
or	
FE	Unit is currently in use.
FE	Operation aborted by user.
FE	xxxx—Unit diagnostics failed.[1]
FE	xxxx—Operation failed.[2]

[1]Refer to the diagnostic error list at the end of this chapter.

[2]xxxx = one of the following error codes:
000D : Cannot write the RCT.
000E : Cannot read the RCT.
000F : Cannot find an RBN to revector to.
0010 : The RAM copy of the bad block table is full.

If a failure is detected, the message indicating the failure will be followed by one or more messages containing error codes.

4.8.5 PARAMS

The PARAMS local program supports modifications to device parameters that you may need to change, such as device node name and allocation class. You invoke it in the same way as the other local programs. However, you use the following commands to make the modifications you need:

EXIT	Terminates PARAMS program
HELP	Prints a brief list of commands and their syntax
SET	Sets a parameter to a value
SHOW	Displays a parameter or a class of parameters
STATUS	Displays module configuration, history, or current counters, depending on the status type chosen
WRITE	Alters the device parameters

4.8.5.1 EXIT

Use the EXIT command to terminate the PARAMS local program.

4.8.5.2 HELP

Use the HELP command to display a brief list of available PARAMS commands, as shown in the example below.

```
PARAMS> HELP
EXIT
HELP
SET {parameter | .} value
SHOW {parameter | . | /class}
  /ALL      /CONST    /DRIVE
  /SERVO    /SCS      /MSCP
  /DUP
STATUS [type]
  CONFIG    LOGS      DATALINK
  PATHS
WRITE

PARAMS>
```

4.8.5.3 SET

Use the SET command to change the value of a given parameter. *Parameter* is the name or abbreviation of the parameter to be changed. To abbreviate, use the first matching parameter without regard to uniqueness. *Value* is the value assigned to the parameter.

For example, SET NODE SUSAN sets the NODENAME parameter to SUSAN.

The following parameters are useful:

ALLCLASS	The controller allocation class. The allocation class should be set to match that of the host.
FIVEDIME	True (1) if MSCP should support five connections with ten credits each. False (0) if MSCP should support seven connections with seven credits each.
UNITNUM	The MSCP unit number.
FORCEUNI	True (1) if the unit number should be taken from the DSSI ID. False (0) if the UNITNUM value should be used instead.
NODENAME	The controller's SCS node name.
FORCENAM	True (1) if the SCS node name should be forced to the string RF31x (where x is a letter from A to H corresponding to the DSSI bus ID) instead of using the NODENAME value. False (0) if NODENAME is to be used.
SYSTEMID	The SYSTEMID parameter provides a number that uniquely identifies the ISE to the operating system. This parameter is modified only when replacing an ISE. Only Customer Services representatives and qualified self-maintenance customers can remove an ISE.

4.8.5.4 SHOW

Use the SHOW command to display the settings of a parameter or a class of parameters. It displays the full name of the parameter (8 characters or less), the current value, the default value, radix and type, and any flags associated with each parameter.

4.8.5.5 STATUS

Use the STATUS command to display module configuration, history, or current counters, depending on the type specified. *Type* is the optional ASCII string that denotes the type of display desired. If you omit *Type*, all available status information is displayed. If present, it may be abbreviated. The following types are available.

CONFIG	Displays the module name, node name, power-on hours, power cycles, and other such configuration information. Unit failures are also displayed, if applicable.
LOGS	Displays the last eleven machine and bug checks on the module. The display includes the processor registers (D0–D7, A0–A7), the time and date of each failure, and some of the hardware registers.
DATALINK	Displays the data link counters.
PATHS	Displays available path information (open virtual circuits) from the point of view of the controller. The display includes the remote node names, DSSI IDs, software type and version, and counters for the messages and datagrams sent and/or received.

4.8.5.6 WRITE

Use the WRITE command to write the changes made while in PARAMS to the ISE's nonvolatile memory. The WRITE command is similar to the VMS SYSGEN WRITE command. Parameters are not available, but you must be aware of the system and/or ISE requirements and use the WRITE command accordingly or it may not succeed in writing the changes.

The WRITE command may fail for one of the following reasons:

- You altered a parameter that required the unit, and the unit cannot be acquired (that is, the unit is not in the available state with respect to the host). Changing the unit number is an example of a parameter that requires the unit.
- You altered a parameter that required a controller initialization, and you replied negatively to the request for reboot. Changing the node name or the allocation class are examples of parameters that require controller initialization.
- Initial ISE calibrations were in progress on the unit. The use of the WRITE command is inhibited while these calibrations are running.

4.9 Diagnostic Error Codes

Diagnostic error codes appear when you are running DRVTST, DRVEXR, or PARAMS. Most of the error codes indicate a failure of the ISE module. The exceptions are listed below. The error codes are listed in Table 4–15. If you see any error code other than those listed below, replace the module.

Table 4–15: ISE Diagnostic Error Codes

Code	Message	Meaning
2032/A032	Failed to see FLT go away	FLT bit of the spindle control status register was asserted for one of the following reasons: 1. Reference clock not present 2. Stuck rotor 3. Bad connection between HDA and module
203A/A03A	Cannot spin up, ACLOW is set in WrtFlt	Did not see ACOK signal, which is supplied by the host system power supply for staggered spin-up.
1314/9314	Front panel is broken	Could be either the module or the operator control panel or both.

Appendix A
KA660 CPU Address Assignments

This appendix lists the CPU address assignments in general and detail for various aspects of memory.

A.1 KA660 Physical Address Space

Table A–1 lists general address assignments for VAX memory and I/O space.

Table A–1: General Local Address Space Map

Address Range	Description
VAX Memory Space	
0000 0000 - 1FFF FFFF	Local Memory Space (512 Mbytes)
VAX I/O Space	
2000 0000 - 2000 1FFF	Local Q22-Bus I/O Space (8 Kbytes)
2000 2000 - 2003 FFFF	Reserved Local I/O Space (248 Kbytes)
2004 0000 - 2007 FFFF	Local ROM Space
2008 0000 - 201F FFFF	Local Register I/O Space (1.5 Mbytes)
2020 0000 - 23FF FFFF	Reserved Local I/O Space (62.5 Mbytes)
2400 0000 - 27FF FFFF	Reserved Local I/O Space (64 Mbytes)
2008 0000 - 2BFF FFFF	Reserved Local I/O Space (64 Mbytes)
2C08 0000 - 2FFF FFFF	Reserved Local I/O Space (64 Mbytes)
3000 0000 - 303F FFFF	Local Q22-Bus Memory Space (4 Mbytes)
3040 0000 - 33FF FFFF	Reserved Local I/O Space (60 Mbytes)
3400 0000 - 37FF FFFF	Reserved Local I/O Space (64 Mbytes)
3800 0000 - 3BFF FFFF	Reserved Local I/O Space (64 Mbytes)
3C00 0000 - 3FFF FFFF	Reserved Local I/O Space (64 Mbytes)

A.2 KA660 Detailed Physical Address Map

Table A–2 lists detailed address assignments for VAX memory and I/O space.

Table A–2: Detailed Local Address Space Map

Description	Address Range
VAX Memory Space	
Local Memory Space, 64 Mbytes (Q22-bus Map at top 32 Kbytes of Main Memory)	0000 0000 - 03FF FFFF
Reserved Memory Space (448 Mbytes)	0400 0000 - 1FFF FFFF
Local Q22-bus I/O Space	**2000 0000 - 2000 1FFF**
Reserved Q22-bus I/O Space	2000 0000 - 2000 0007
Q22-bus Floating Address Space	2000 0008 - 2000 07FF
User Reserved Q22-bus I/O Space	2000 0800 - 2000 0FFF
Reserved Q22-bus I/O Space	2000 1000 - 2000 1F3F
Interprocessor Comm Reg	2000 1F40
Reserved Q22-bus I/O Space	2000 1F44 - 2000 1FFF
Local Register I/O Space	**2000 2000 - 2003 FFFF**
Reserved Local Register I/O Space	2000 4202 - 2000 422F
SHAC SSWCR	2000 4230
Reserved Local Register I/O Space	2000 4234 - 2000 4043
SHAC SSHMA	2000 4244
SHAC PQBBR	2000 4248
SHAC PSR	2000 424C
SHAC PESR	2000 4250
SHAC PFAR	2000 4254
SHAC PPR	2000 4258

Table A–2 (Cont.): Detailed Local Address Space Map

Description	Address Range
Local Register I/O Space	**2000 2000 - 2003 FFFF**
SHAC PMCSR	2000 425C
Reserved Local Register I/O Space	2000 4260 - 2000 427F
SHAC PCQ0CR	2000 4280
SHAC PCQ1CR	2000 4284
SHAC PCQ2CR	2000 4288
SHAC PCQ3CR	2000 428C
SHAC PDFQCR	2000 4290
SHAC PMFQCR	2000 4294
SHAC PSRCR	2000 4298
SHAC PECR	2000 429C
SHAC PDCR	2000 42A0
SHAC PICR	2000 42A4
SHAC PMTCR	2000 42A8
SHAC PMTECR	2000 42AC
Reserved Local Register I/O Space	2000 42B0 - 2000 7FFF
NICSR0 - Vector Add, IPL, Sync/Async	2000 8000
NICSR1 - Polling Demand Register	2000 8004
NICSR2 - Reserved	2000 8008
NICSR3 - Receiver List Address	2000 800C
NICSR4 - Transmitter List Address	2000 8010
NICSR5 - Status Register	2000 8014
NICSR6 - Command and Mode Register	2000 8018
NICSR7 - System Base Address	2000 801C

Table A–2 (Cont.): Detailed Local Address Space Map

Description	Address Range
Local Register I/O Space	**2000 2000 - 2003 FFFF**
NICSR8 - Reserved	2000 8020+
NICSR9 - Watchdog Timers	2000 8024+
NICSR10- Reserved	2000 8028+
NICSR11- Rev Num & Missed Frame Count	2000 802C+
NICSR12- Reserved	2000 8030+
NICSR13- Breakpoint Address	2000 8034+
NICSR14- Reserved	2000 8038+
NICSR15- Diagnostic Mode & Status	2000 803C
Reserved Local Register I/O Space	2000 8040 - 2003 FFFF
Local EPROM I/O Space	**2004 0000 - 2007 FFFF**
μVAX System Type Register (In EPROM)	2004 0004
Local EPROM - (Halt Protected)	2004 0000 - 2007 FFFF
Local Register I/O Space	**2008 0000 - 201F FFFF**
Q22 System Configuration Register	2008 0000
Q22 System Error Register	2008 0004
Q22 Master Error Address Register	2008 0008
Q22 Slave Error Address Register	2008 000C
Q22-bus Map Base Register	2008 0010
Reserved Local Register I/O Space	2008 0014 - 2008 00FF
Main Memory Error Status Register	2008 0140
Main Memory Control/Diag Status Register	2008 0144
Reserved Local Register I/O Space	2008 0148 - 2008 3FFF

Table A–2 (Cont.): Detailed Local Address Space Map

Description	Address Range
Local Register I/O Space	**2008 0000 - 201F FFFF**
Boot and Diagnostic Reg (32 Copies)	2008 4000 - 2008 407C
Reserved Local Register I/O Space	2008 4080 - 2008 7FFF
Q22-bus Map Registers	2008 8000 - 2008 FFFF
Reserved Local Register I/O Space	2009 0000 - 2013 FFFF
SSC Base Address Register	2014 0000
SSC Configuration Register	2014 0010
CDAL Bus Timeout Control Register	2014 0020
Diagnostic LED Register	2014 0030
Reserved Local Register I/O Space	2014 0034 - 2014 006B

NOTE: *The following addresses allow those KA660 internal processor registers that are implemented in the SSC chip (external, internal processor registers) to be accessed via the local I/O page. These addresses are documented for diagnostic purposes only and should not be used by non-diagnostic programs.)*

Time Of Year Register	2014 006C
Console Storage Receiver Status	2014 0070*
Console Storage Receiver Data	2014 0074*
Console Storage Transmitter Status	2014 0078*
Console Storage Transmitter Data	2014 007C*
Console Receiver Control/Status	2014 0080
Console Receiver Data Buffer	2014 0084
Console Transmitter Control/Status	2014 0088

Table A–2 (Cont.): Detailed Local Address Space Map

Description	Address Range
Console Transmitter Data Buffer	2014 008C
Reserved Local Register I/O Space	2014 0090 - 2014 00DB
I/O Bus Reset Register	2014 00DC
Reserved Local Register I/O Space	2014 00E0
Rom Data Register	2014 00F0**
Bus Timeout Counter	2014 00F4**
Interval Timer	2014 00F8**
Reserved Local Register I/O Space	2014 00FC - 2014 00FF
Timer 0 Control Register	2014 0100
Timer 0 Interval Register	2014 0104
Timer 0 Next Interval Register	2014 0108
Timer 0 Interrupt Vector	2014 010C
Timer 1 Control Register	2014 0110
Timer 1 Interval Register	2014 0114
Timer 1 Next Interval Register	2014 0118
Timer 1 Interrupt Vector	2014 011C
Reserved Local Register I/O Space	2014 0120 - 2014 012F
BDR Address Decode Match Register	2014 0130
BDR Address Decode Mask Register	2014 0134
Reserved Local Register I/O Space	2014 0138 - 2014 03FF
Battery Backed-Up RAM	2014 0400 - 2014 07FF

Table A–2 (Cont.): Detailed Local Address Space Map

Description	Address Range
Reserved Local Register I/O Space	2014 0800 - 201F FFFF
Reserved Local I/O Space	2020 0000 - 2FFF FFFF
Local Q22-bus Memory Space	3000 0000 - 303F FFFF
Reserved Local Register I/O Space	3040 0000 - 3FFF FFFF

A.3 External and Internal Processor Registers

Several of the internal processor registers (IPR's) on the KA660 are implemented in the SSC chip rather than the SOC CPU chip. These registers are referred to as external and internal processor registers and are listed below.

Table A–3: External, Internal Processor Registers

IPR #	Register Name	Mnemonic
27	Time of Year Register	TOY
28	Console Storage Receiver Status	CSRS*
29	Console Storage Receiver Data	CSRD*
30	Console Storage Transmitter Status	CSTS*
31	Console Storage Transmitter Data	CSDB*
32	Console Receiver Control/Status	RXCS
33	Console Receiver Data Buffer	RXDB
34	Console Transmitter Control/ Status	TXCS
35	Console Transmitter Data Buffer	TXDB
55	I/O System Reset Register	IORESET

A.4 Global Q22-Bus Physical Address Space

Table A–4 lists the global Q22-bus physical address map.

Table A–4: Global Q22-bus Physical Address Map

Description	Address Range
Q22-bus Memory Space	
Q22-bus Memory Space (Octal)	0000 0000 - 1777 7777
Q22-bus I/O Space (BBS7 Asserted)	
Q22-bus I/O Space (Octal)	**1776 0000 - 1777 7777**
Reserved Q22-bus I/O Space	1776 0000 - 1776 0007
Q22-bus Floating Address Space	1776 0010 - 1776 3777
User Reserved Q22-bus I/O Space	1776 4000 - 1776 7777
Reserved Q22-bus I/O Space	1777 0000 - 1777 7477
Interprocessor Comm Reg	1777 7500
Reserved Q22-bus I/O Space	1777 7502 - 1777 7777

Appendix B
Programming Parameters for RF-Series ISEs

This appendix describes the procedures for setting and examining parameters for RF-series ISEs.

Two types of DSSI storage adapters are available for VAX 4000, MicroVAX 3000-series, MicroVAX II, and DECsystem systems: an embedded DSSI host adapter that is part of the CPU and the KFQSA storage adapter.

Each storage adapter provides a separate DSSI bus that can support up to seven RF-series ISEs (six ISEs for a dual-host configuration). The adapters make a connection between the CPU and the requested ISE on their respective DSSI bus. Each ISE has its own controller and server that contain the intelligence and logic necessary to control data transfers over the DSSI bus.

B.1 RF-Series ISE Parameters

Six principal parameters are associated with each RF-series ISE:

- Bus Node ID
- ALLCLASS
- UNITNUM
- FORCEUNI
- NODENAME
- SYSTEMID

NOTE: *Each of the above ISE parameters, with the exception of the Bus Node ID, are programmed and examined using the console-based Diagnostic and Utility Protocol (DUP) driver utility. The ISE Bus Node ID is physically determined by the numbered bus node ID plug that inserts into the ISE front panel.*

A brief description of each parameter follows:

The Bus Node ID parameter is provided by the bus node ID plug on the ISE front panel. Each DSSI bus can support up to seven ISEs, bus nodes 0 through 6 (0 through 5 for dual-host systems). Refer to your *Operation* manual for instructions on changing bus node ID plugs.

The ALLCLASS parameter determines the device allocation class. The allocation class is a numeric value from 0 to 255 that is used by the VMS operating system to derive a path-independent name for multiple access paths to the same ISE. RF-series ISEs are shipped from the factory with a default allocation class of zero. Each RF-series ISE to be served to the cluster should have an allocation class that matches the allocation class of the host system. Refer to the *VMS VAXcluster* manual for rules for specifying allocation class values.

The UNITNUM parameter determines the unit number of the ISE. By default, the ISE unit number is supplied by the Bus Node ID plug on the ISE front panel. Certain multiple bus configurations, described later on in this section, require that the default values be replaced with unique ISE unit numbers. To set unit numbers and override the default values, you use the console-based DUP driver utility to supply values to the UNITNUM parameter and to set a value of zero to ISE parameter FORCEUNI.

The FORCEUNI parameter controls the use of UNITNUM to override the default ISE unit number supplied by the Bus Node ID plug. When FORCEUNI is set to a value of zero, the operating system uses the value assigned to the UNITNUM parameter; when FORCEUNI is set to a value of one, the operating system uses the value supplied by the Bus Node ID plug.

The NODENAME parameter allows each ISE to have an alphanumeric node name of up to eight characters. RF-series ISEs are shipped from the factory with a unique identifier, such as R7CZZC, R7ALUC, and so on. You can provide a node name of your choosing if you prefer.

The SYSTEMID parameter provides a number that uniquely identifies the ISE to the operating system. This parameter is modified only when replacing an ISE. Only Customer Services representatives and qualified self-maintenance customers can remove an ISE.

The following describes how the operating system uses the ISE parameters to form unique identifiers for each ISE. Configurations that require you to assign new unit numbers for ISEs are also described.

With an allocation class of zero, the operating system can use the default parameter values to provide each ISE with a unique device name. The operating system uses the node name along with the device logical name in the following manner:

NODENAME$DIA*u*

where:

NODENAME is a unique node name and *u* is the unit number.

With a nonzero allocation class, the operating system relies on unit number values to create a unique device name. The operating system uses the allocation class along with the device logical name in the following manner:

$ALLCLASS$DIA*u*

where:

ALLCLASS is the allocation class for the system and ISEs, and *u* is a unique unit number.

Using the KFQSA storage adapter and mass storage expanders, you can fill multiple DSSI busses. Each bus can have seven ISEs (bus nodes 0–6). When a second bus is added to the system, and your system is using a nonzero allocation class, you need to assign new unit numbers for ISEs on one of the busses, as the unit numbers for ISEs throughout the system must be unique. Table B–1 illustrates the need to program unit numbers for a system using both more than one DSSI bus and a nonzero allocation class. In the case of the nonzero allocation class, the operating system sees the ISEs as having duplicate device names.

Table B–1: How the VMS Operating System Identifies the ISEs

Allocation Class=0	Nonzero Allocation Class (Example; ALLCLASS=1)
R7CZZC$DIA0	1DIA0*
R7ALUC$DIA1	1DIA1*
R7EB3C$DIA2	1DIA2*
R7IDFC$DIA0	1DIA0*
R7IBZC$DIA1	1DIA1*
R7IKJC$DIA2	1DIA2*
R7ID3C$DIA3	1DIA3
R7XA4C$DIA4	1DIA4
R7QIYC$DIA5	1DIA5
R7DA4C$DIA6	1DIA6

*Nonzero allocation class examples with an asterisk indicate duplicate device names. For one of the DSSI busses, the unit numbers need to be reprogrammed to avoid this error.

The following instructions describe how to change ISE parameters using the DUP driver utility. In the sample procedures, the allocation class will be set to 2, the ISEs will be assigned new unit numbers, and the system disk will be assigned a new node name.

1. Enter the console mode.

   The procedure for programming internal parameters for RF-series ISEs requires that you issue commands to those RF-series ISEs at the console prompt (>>>). You may enter these commands in either uppercase or lowercase letters. Unless otherwise instructed, enter each command, then press Return.

   Enter console mode as follows:

   a. Set the Break Enable/Disable switch on the CPU cover panel to the enable position.

   b. Set the power switch for each unit (both hosts for a dual-host system, and any expanders for expanded systems) to on (1).

   Wait for the system to display the console prompt (>>>).

2. Make sure the ISEs for which you want to set parameters are on line and are not write protected. The Run/Ready button should be (lit), and the Write-Protect button should be out (not lit).

3. For systems with embedded DSSI, enter `SHOW DSSI` at the console prompt for a display of all DSSI devices in your expanded system. For KFQSA-based DSSI, enter `SHOW UQSSP`.

   The firmware displays two lines of information for each ISE. The first line contains the node number and node name. The second line contains the device name and unit number followed by the device type in parentheses.

   For embedded DSSI, the device name consists of the letters DIA*n* and the DSSI host adapter is identified by an asterisk (*). For KFQSA-based DSSI, the device name consists of the letters DU*cn*, where *c* is the controller letter, and *n* is a unique unit number.

The following examples show a system with three RF31 ISEs. Example B–1 shows a system with embedded DSSI and Example B–2 shows a system with KFQSA-based DSSI.

Example B–1: SHOW DSSI Display (Embedded DSSI)

```
>>>SHOW DSSI
DSSI Bus 0 Node 0 (R7CZZC)
-DIA0 (RF31)
DSSI Bus 0 Node 1 (R7ALUC)
-DIA1 (RF31)
DSSI Bus 0 Node 2 (R7EB3C)
-DIA2 (RF31)
DSSI Bus 0 Node 7 (*)
>>>
```

Example B–2: SHOW UQSSP Display (KFQSA-Based DSSI)

```
>>>SHOW UQSSP
UQSSP Disk Controller 0 (772150)
-DUA0 (RF31)
UQSSP Disk Controller 1 (760334)
-DUB1 (RF31)
UQSSP Disk Controller 2 (760340)
-DUC2 (RF31)
UQSSP Tape Controller 0 (774500)
-MUA0 (TK70)
```

In this example, each ISE will be assigned an allocation class of 2, and the system disk will be given a new node name. Also, ISEs DIA0, DIA1, and DIA2 (or DUA0, DUB1, and DUC2) will be assigned unit numbers 10, 11, and 12, respectively.

B.2 Entering the DUP Driver Utility

To examine and change internal RF-series ISE parameters, you must first activate the DUP driver utility by setting host to the specific ISE for which you want to modify or examine parameters.

Use the following command for embedded DSSI:

SET HOST/DUP/DSSI <node_number> PARAMS

where:

<node_number> is the bus node ID (0–6) for the ISE on the bus.

Use the following command for KFQSA-based DSSI:

SET HOST/DUP/UQSSP/DISK <node_number> PARAMS

where:

<node_number> is the bus node ID (0–6) for the ISE on the bus.

The following examples show the commands entered at the console prompt to start the DUP server for the ISE at node 0. In Example B–3, you enter `SET HOST/DUP/DSSI 0 PARAMS` for embedded DSSI. In Example B–4, you enter `SET HOST/DUP/UQSSP/DISK 0 PARAMS` for KFQSA-based DSSI.

Example B–3: Starting the DUP Driver Utility (Embedded DSSI)

```
>>>SET HOST/DUP/DSSI 0 PARAMS
Starting DUP server...
Copyright (c) 1990  Digital Equipment Corporation

PARAMS>
```

Example B–4: Starting the DUP Driver Utility (KFQSA-Based DSSI)

```
>>>SET HOST/DUP/UQSSP/DISK 0 PARAMS
Starting DUP server...
Copyright (c) 1990  Digital Equipment Corporation

PARAMS>
```

B.3 Setting Allocation Class

After entering the DUP driver utility for a specified ISE, you can examine and set the allocation class for the ISE as follows:

1. At the PARAMS> prompt, enter SHOW ALLCLASS to check the allocation class of the ISE to which you are currently connected.
2. Enter SET ALLCLASS 2 (or enter the allocation class you desire).
3. Enter SHOW ALLCLASS to verify the new allocation class.

Example B–5 shows the steps for examining and changing the allocation class for a specified ISE. In the example, the allocation class is changed from an allocation class of 0 to an allocation class of 2.

Example B–5: Setting Allocation Class for a Specified ISE

```
PARAMS>SHOW ALLCLASS
Parameter      Current           Default         Type      Radix
---------  ----------------  ----------------  --------  -----
ALLCLASS                 0                 0      Byte      Dec    B

PARAMS>SET ALLCLASS 2
PARAMS>SHOW ALLCLASS
Parameter      Current           Default         Type      Radix
---------  ----------------  ----------------  --------  -----
ALLCLASS                 2                 0      Byte      Dec    B
```

B.4 Setting Unit Number

After entering the DUP driver utility for a specified ISE, you can examine and set the unit number for the ISE as follows:

1. At the PARAMS> prompt, enter SHOW UNITNUM to check the unit number of the ISE to which you are currently connected.
2. Enter SET UNITNUM 10 (or enter the unit number you desire).
3. Enter SET FORCEUNI 0 to override the default unit number value supplied by the bus node ID plug.
4. Enter SHOW UNITNUM to verify the new unit number.
5. Enter SHOW FORCEUNI to verify that the current value for the FORCEUNI parameter is 0.

   Example B–6 shows the steps for changing the unit number of a specified ISE from unit number 0 to unit number 10.
6. Label the ISE with its unit number, using the unit number labels shipped with your system. Figure B–1 shows where to affix a unit number label on the ISE front panel.

Example B–6: Setting a Unit Number for a Specified ISE

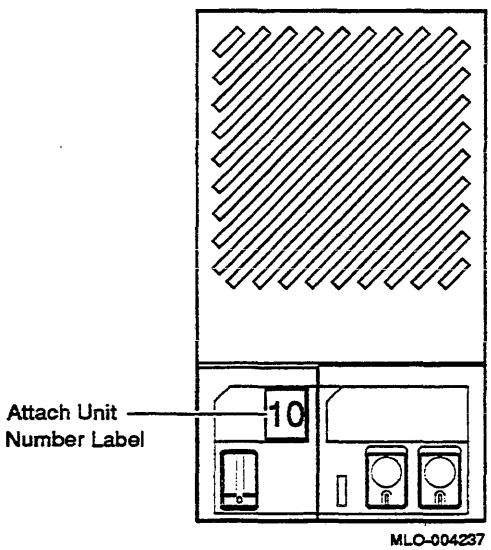
```
PARAMS>SHOW  UNITNUM
Parameter      Current          Default          Type      Radix
---------  ---------------  ---------------  --------  -----
UNITNUM                0                0      Word      Dec    U

PARAMS>SET UNITNUM 10
PARAMS>SET FORCEUNI 0
PARAMS>SHOW UNITNUM

Parameter      Current          Default          Type      Radix
---------  ---------------  ---------------  --------  -----
UNITNUM               10                0      Word      Dec    U

PARAMS>SHOW FORCEUNI
Parameter      Current          Default          Type      Radix
---------  ---------------  ---------------  --------  -----
FORCEUNI               0                1   Boolean      0/1    U
```

Figure B–1: Attaching a Unit Number Label to the ISE Front Panel



B.5 Setting Node Name

After entering the DUP driver utility for a specified ISE, you can examine and set the node name for the ISE as follows:

1. At the PARAMS> prompt, enter SHOW NODENAME to check the node name of the ISE to which you are currently connected.
2. Enter SET NODENAME SYSDSK (or enter the desired alphanumeric node name of up to eight characters).
3. Enter SHOW NODENAME to verify the new node name.

Example B–7 shows the steps for changing the node name of a specified ISE from the factory-supplied name to SYSDSK.

Example B–7: Changing a Node Name for a Specified ISE

```
PARAMS>SHOW NODENAME
Parameter      Current            Default         Type      Radix
---------  ----------------  ----------------  --------  -----
NODENAME          R7CZZC             RF31        String  Ascii   B

PARAMS>SET NODENAME SYSDSK

PARAMS>SHOW NODENAME
Parameter      Current            Default         Type      Radix
---------  ----------------  ----------------  --------  -----
NODENAME          SYSDSK             RF31        String  Ascii   B
```

B.6 Setting System ID

NOTE: *This parameter is modified only when replacing an ISE. Only Customer Services representatives and qualified self-maintenance customers should remove an ISE. All parameters for the replacement ISE should be programmed to match those of the original ISE. When replacing a ISE, be sure to set the SYSTEMID parameter to match the that of the original.*

After entering the DUP driver utility for a specified ISE, you can examine and set the system ID for the ISE as follows:

1. At the PARAMS> prompt, enter SHOW SYSTEMID to check the system ID of the ISE to which you are currently connected.
2. Enter SET SYSTEMID *System ID* (enter the desired serial number-based system ID).

3. Enter SHOW SYSTEMID to verify the new system ID.

Example B–8 shows the steps for changing the system ID of a specified ISE from the factory-supplied system ID to 1402193310841 (the system ID for the replacement ISE is programmed to match that of the original ISE).

Example B–8: Changing a System ID for a Specified ISE

```
PARAMS>SHOW SYSTEMID
Parameter      Current           Default          Type      Radix
---------  ----------------  ----------------  --------  -----
SYSTEMID     0402193310841     0000000000000  Quadword     Hex   B

PARAMS>SET SYSTEMID 1402193310841

PARAMS>SHOW SYSTEMID
Parameter      Current           Default          Type      Radix
---------  ----------------  ----------------  --------  -----
SYSTEMID     1402193310841     0000000000000  Quadword     Hex   B
```

B.7 Exiting the DUP Server Utility

After you have completed setting and examining internal ISE parameters, enter the WRITE command at the PARAMS> prompt to save the ISE parameters you have changed using the SET command. The changes are recorded to nonvolatile memory.

If you have changed the allocation class or node name of an ISE, the DUP driver utility will ask you to initialize the controller. Answer Yes (Y) to allow the changes to be recorded and to exit the DUP driver utility.

If you have not changed the allocation class or node name, enter the EXIT command at the PARAMS> prompt to exit the DUP driver utility for the specified ISE. Example B–9 shows the procedure for saving parameter changes. In the example, the controller is initialized.

Example B–9: Exiting the DUP Driver Utility for a Specified ISE

```
PARAMS>WRITE
Changes require controller initialization, ok? [Y/(N)] Y

Stopping DUP server...
>>>
```

NOTE: *You must repeat the procedures in this chapter for each ISE for which you want to change parameters.*

Example B–10 shows the display for the SHOW DSSI command for a system with embedded DSSI after the unit numbers for the ISEs have been changed from 0, 1, and 2 to 10, 11, and 12. Notice that the bus 0 device names are now DIA10, DIA11, and DIA12.

Example B–10: SHOW DSSI Display

```
>>>SHOW DSSI
DSSI Bus 0 Node 0 (SYSDSK)
-DIA10 (RF31)
DSSI Bus 0 Node 1 (R7ALUC)
-DIA11 (RF31)
DSSI Bus 0 Node 2 (R7EB3C)
-DIA12 (RF31)
DSSI Bus 0 Node 7 (*)
>>>
```

Example B–11 shows the display for the SHOW UQSSP command for a system with KFQSA-based DSSI.

Example B–11: SHOW UQSSP Display (KFQSA-Based DSSI)

```
>>>SHOW UQSSP
UQSSP Disk Controller 0 (772150)
-DUA0 (RF31)
UQSSP Disk Controller 1 (760334)
-DUB1 (RF31)
UQSSP Disk Controller 2 (760340)
-DUC2 (RF31)
UQSSP Tape Controller 0 (774500)
-MUA0 (TK70)
```

Index

D

E

F

G

H

I

K

L

M

N

HOW TO ORDER ADDITIONAL DOCUMENTATION

From	Call	Write
Alaska, Hawaii, or New Hampshire	603–884–6660	Digital Equipment Corporation P.O. Box CS2008 Nashua NH 03061
Rest of U.S.A. and Puerto Rico[1]	800–DIGITAL	
[1]Prepaid orders from Puerto Rico, call Digital's local subsidiary (809–754–7575)		
Canada	800–267–6219 (for software documentation)	Digital Equipment of Canada Ltd. 100 Herzberg Road Kanata, Ontario, Canada K2K 2A6 Attn: Direct Order Desk
	613–592–5111 (for hardware documentation)	
Internal orders (for software documentation)	DTN: 241–3023 508–874–3023	Software Supply Business (SSB) Digital Equipment Corporation Westminster MA 01473
Internal orders (for hardware documentation)	DTN: 234–4323 508–351–4323	Publishing & Circulation Services (P&CS) NRO3–1/W3 Digital Equipment Corporation Northboro MA 01532

Reader's Comments

KA660 CPU System Maintenance

EK–398AA–MM–001

Your comments and suggestions help us improve the quality of our publications.

Please rate the manual in the following categories:	Excellent	Good	Fair	Poor
Accuracy (product works as described)	☐	☐	☐	☐
Completeness (enough information)	☐	☐	☐	☐
Clarity (easy to understand)	☐	☐	☐	☐
Organization (structure of subject matter)	☐	☐	☐	☐
Figures (useful)	☐	☐	☐	☐
Examples (useful)	☐	☐	☐	☐
Table of contents (ability to find topic)	☐	☐	☐	☐
Index (ability to find topic)	☐	☐	☐	☐
Page design (overall appearance)	☐	☐	☐	☐
Print quality	☐	☐	☐	☐

What I like best about this manual: _____

_____

What I like least about this manual: _____

_____

Additional comments or suggestions: _____

_____

_____

I found the following errors in this manual:

Page	Description
_____	_____
_____	_____
_____	_____

For which tasks did you use this manual?

☐ Installation
☐ Maintenance
☐ Marketing
☐ Operation/Use
☐ Programming
☐ System Management
☐ Training
☐ Other (please specify) _____

Name/Title _____

Company _____

Address _____

Do Not Tear - Fold Here and Tape

digital

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 33 MAYNARD MASS.

POSTAGE WILL BE PAID BY ADDRESSEE

DIGITAL EQUIPMENT CORPORATION
CORPORATE USER PUBLICATIONS
PKO3-1/D30
129 PARKER STREET
MAYNARD, MA 01754-9975

Do Not Tear - Fold Here and Tape