KA630-A Processor

Specification

Rev. 3.4 Date: 26-Apr-85

P R E L I M I N A R Y

Jay Nichols
DTN 223-7094
ML5-5/E71

Table of Contents

		Page
1.0	Introduction	1-1
	1.1 Scope of Document	1-1
	1.2 Applicable Documents	1-1
2.0	General Description	2-1
	2.1 KA630 and MS630 Option Summary	2-2
	2.2 KA630-A Processor Module Feature Summary	2-3
3.0	Physical Specifications	3-1
	3.1 Dimensions	3-1
	3.2 KA630 Module Pinouts	3-2
	3.3 KA630 Module Connectors	3-2
	3.4 MS630 Module Pinouts	3-5
	3.5 MS630 Module Connectors	3-6
4.0	Electrical Specifications	4-1
	4.1 DC Power Consumption	4-1
	4.2 Bus Loads	4-1
	4.3 DCOK Signal	4-2
	4.4 POK Signal	4-2
	4.5 Battery Backup Specifications	4-2
5.0	Environmental and Reliability	5-1
	5.1 Storage Conditions	5-1
	5.2 Operating Conditions	5-1
	5.3 Mean Time Before Failure	5-1
6.0	Central Processor	6-1
	6.1 MicroVAX CPU Chip Description	6-1
	6.2 Processor State	6-2
	6.2.1 General Purpose Registers	6-2
	6.2.2 Processor Status Longword	6-3
	6.2.3 Processor Registers	6-4
	6.3 Instruction Set	6-8
	6.4 Interrupts and Exceptions	6-8
	6.4.1 Interrupts	6-9
	6.4.2 Exceptions	6-10
	6.4.3 Machine Check Parameters	6-11
	6.4.4 Halt Conditions	6-12
	6.4.5 System Control Block (SCB)	6-13
	6.5 Hardware Detected Errors	6-15
	6.5.1 Non-Existent Memory Errors	6-15
	6.5.2 Parity Error Detection	6-15
	6.5.3 Interrupt Vector Timeouts	6-16
	6.5.4 "No Sack" Timeouts	6-17
	6.6 Latency Specifications	6-17
	6.6.1 Interrupt Latency	6-17
	6.6.2 Interrupt Service Time	6-18
	6.6.3 DMA Latency	6-18
	6.7 System Identification	6-19

7.0 Memory Management 7-1
7.1 Physical and Virtual Address Space 7-1
7.2 Memory Management Control Registers 7-2
7.3 System Space Address Translation 7-2
7.4 Process Space Address Translation 7-4
7.4.1 P0 Region Address Translation 7-4
7.4.2 P1 Region Address Translation 7-6
7.5 Page Table Entry 7-8

8.0 Floating Point 8-1

9.0 KA630 Memory System 9-1
9.1 Memory System Summary 9-1
9.2 KA630 Local Memory 9-1
9.3 Mapping Registers 9-1
9.3.1 Mapping Register Format 9-1
9.3.2 Mapping Register Addresses 9-2
9.3.3 Q22-Bus Map Operation 9-3
9.4 Memory System Registers 9-4
9.4.1 Memory System Error Register 9-4
9.4.2 CPU Error Address Register 9-8
9.4.3 DMA Error Address Register 9-8
9.5 Memory System Operation 9-9

10.0 KA630 Boot and Diagnostic Facility 10-1
10.1 Boot and Diagnostic Register 10-1
10.2 ROM Memory 10-3
10.2.1 ROM Sockets 10-3
10.2.2 ROM Address Space 10-3
10.2.3 KA630-A Console Program Operation 10-4

11.0 Time of Year Clock 11-1
11.1 Battery Backed-up Watch Chip 11-1
11.2 Watch Chip Registers 11-1
11.2.1 Time of Year Data Registers 11-2
11.2.2 Control and Status Register A 11-2
11.2.3 Control and Status Register B 11-3
11.2.4 Control and Status Register C 11-4
11.2.5 Control and Status Register D 11-4
11.2.6 RAM Memory 11-4
11.3 Powerup 11-5
11.3.1 Valid RAM and Time 11-5
11.3.2 Invalid RAM and Time 11-5

12.0 Interval Timer 12-1
12.1 Interval Clock Control/Status Register 12-1
12.2 Interval Timer Operation 12-1

13.0 Console Serial Line 13-1
13.1 Console Functionality 13-1
13.2 Console Registers 13-1
13.2.1 Console Receiver C/S Register 13-2
13.2.2 Console Receiver Data Buffer 13-3
13.2.3 Console Transmitter C/S Register 13-4
13.2.4 Console Transmitter Data Buffer 13-5
13.3 Additional Specifications 13-5
13.4 Break Response 13-5

14.0 Q22-Bus Control 14-1
14.1 Bus Initialize Register 14-1
14.2 Multi-level Interrupts 14-1
14.3 Interprocessor Communication Facility 14-1
14.3.1 Interprocessor Comm Register 14-2
14.3.2 Interprocessor Doorbell Interrupts 14-3

15.0 Multi-Processor Considerations 15-1
15.1 Auxiliary/Arbiter Differences 15-1
15.2 Multi-Processor Features 15-2
15.3 KA630 Based Multi-Processor Systems 15-2
15.4 PDP11 Based Multi-Processor Systems 15-3

Appendix A Module Pinouts A-1

Appendix B Connector and Jumper Summary B-1

Appendix C Physical Address Assignments C-1

REVISION HISTORY

REV	DATE	REASON
0.0	27-Jun-83	Working Document: For use by Mayflower Design Team.
1.0	02-Aug-83	Preliminary Spec: Sent out for review.
2.0	01-Nov-83	The more significant changes include: 1) the change from memory daughter boards to memory expansion modules; 2) expansion of Q22-Bus map to 4MB; 3) the change in the handling of multi-level Q22-Bus interrupts, including the elimination of the Interrupt Control Register; 4) further simplification of the multi-processor hooks and arbiter/auxiliary differences 5) Changes to the Memory System Error and Memory Error Address Registers and elimination of the PDP-11 compatible Memory Control and Status Register.
3.0	20-Feb-84	The more significant changes include: 1) The 24-pin external connector has become two connectors: a 10-pin console SLU connector and a 20-pin Display and Configuration Connector. 2) Memory expansion signal pinouts for the CD Interconnect have been defined. 3) The Memory Error Address Register has been split out into two registers: the CPU Error Address Register and the DMA Error Address Register. 4) The definition of Memory System Error Register bit <00> (PAR ENB) has been clarified. 5) The number of BDG Code bits has been reduced from three to two, affecting both the Boot and Diagnostic Register and the Display and Configuration Connector.
3.1	20-Jul-84	The more significant changes include: 1) The names KDQ32 and MSA32 have been changed to KA630 and MS630 respectively. 2) The four error display bits in the Boot and Diagnostic Register are now read/write bits (section 10.1). 3) The battery backup specification for the Time of Year Clock is now 240 hours.

REV	DATE	REASON
3.2	17-Oct-84	The more significant changes include: 1) Console SLU Connector Pin 01 has been defined (section 3.3.3). 2) Section added on system identification registers (section 6.7). 3) Memory Arbiter gives higher priority to external device requests (section 9.5). 4) Note added on treatment of address bit 0 during byte reads (section 9.5). 5) The setup of TOY Clock CSR A has been corrected (section 11.2.1).
3.3	04-Mar-85	The more significant changes include: 1) The DC Power AC/DC loading values have been added (sections 4.1 and 4.2). 2) Mean Time between Failure specifications have been updated (section 5.3). 3) The Local Memory External Access Enable bit in the Interprocessor Communication Register is now cleared by writes to Bus Initialize Register; reflects design, which can not be changed (section 14.3.1).
3.3	04-Mar-85	The more significant changes include: 1) The maximum operating temperature was reduced to reflect characteristics of the current version of MicroVAX chip (section 5.2).

1.0 Introduction

1.1 Scope of Document

This specification documents the functional, physical and environmental characteristics of the KA630-A Processor Module. It also provides information on the MS630 memory expansion modules which are documented more completely in the MS630 Memory Module Specification.

1.2 Applicable Documents

The following reference material contains detailed information regarding the VAX-11 family, the KA630-A module, and the required environment.

KA630-A Console Program Specification

MS630 Memory Module Specification

MicroVAX Architecture Reference Manual

MicroVAX CPU Chip Engineering Specification
MicroVAX CPU Chip Design Specification
MicroVAX FPU Chip Engineering Specification
MicroVAX FPU Chip Design Specification

VAX Architecture Handbook
VAX Software Handbook
VAX Hardware Handbook

DEC STD 32	VAX Architecture Standard
DEC STD 102	Environmental Specification
DEC STD 160	Q-Bus Specification

2.0 General Description

The KA630 is a quad height Q22-Bus VAX11 which consists of a single KA630 processor module.

The KA630 module contains a MicroVAX Central Processor Chip, which includes Memory Management, an optional Floating Point Processor Chip, either 256KB or 1MB of on-board memory, a Q22-Bus interface, a Q22-Bus Map for DMA transfers, an Interval Timer, a Boot and Diagnostic Facility, a console serial line unit and a time of year clock with support for battery backup (batteries are located on the system back panel).

The KA630 module utilizes the backplane CD Interconnect and a 50-pin connector to communicate with up to two MS630 memory expansion modules, each of which may contain 1MB, 2MB, 4MB or 8MB of additional local memory.

The KA630 module can be configured as the arbiter CPU or as one of three auxiliary CPU's. An interprocessor communication register facilitates the use of this module in multiprocessor systems.

When configured as the arbiter CPU, the KA630 mounts in the first slot of a Q22-Bus backplane. This first backplane slot, plus any slots occupied by MS630 modules, must feature the CD Interconnect. The arbiter KA630 arbitrates Bus Mastership and fields Q22-Bus interrupt requests BR7-4. It can also respond to interrupt requests from its own interval timer, console serial line unit and interprocessor doorbell.

When configured as an auxiliary CPU, the KA630 mounts in any Q22-Bus/CD-Interconnect backplane slot not already occupied by the arbiter CPU or associated memory expansion modules. The arbiter may be a Q22-Bus PDP11 or another KA630. The auxiliary KA630 requests bus mastership to access the Q22-Bus. It does not field Q22-Bus interrupt requests, but can respond to interrupt requests from its own interval timer, console serial line unit and interprocessor doorbell.

2.1 KA630 and MS630 Option Summary

The KA630-A processor module is available with either 256KB or 1MB of on-board memory and with or without the floating point processor. The option designations are as follows:

Option	Description
KA630-AA	Quad height Q22-Bus MicroVAX CPU Module includes MicroVAX processor chip, floating point processor chip, 1MB on-board memory, Console SLU, Interval Timer, Boot/Diag ROM, and Q22-Bus Map/Interface; accepts up to two MS630 memory expansion modules; can be configured as Arbiter or Auxiliary CPU.
KA630-AB	KA630-AA without floating point processor chip.
KA630-AC	KA630-AA with only 256KB of on-board memory.
KA630-AD	KA630-AC without floating point processor chip.

The MS630 memory expansion modules are available with 1MB, 2MB, 4MB or 8MB of memory. The option designations are as follows:

Option	Description
MS630-AA	1MB Memory Expansion Module for KA630-A Processor; dual height module with 256K RAM's.
MS630-BA	2MB Memory Expansion Module for KA630-A Processor; (half populated MS630-BB).
MS630-BB	4MB Memory Expansion Module for KA630-A Processor; quad height module with 256K RAM's.
MS630-CA	8MB Memory Expansion Module for KA630-A Processor; quad height module with 1M RAM's.

2.2 KA630-A Processor Module Feature Summary

KA630-A Processor Module Summary

- Quad Height Module

- Configurable as Arbiter or Auxiliary CPU
  - Each System contains one Arbiter CPU
  - Each System may contain up to three Auxiliary CPU's

- MicroVAX Processor Chip
  - MicroVAX Subset of VAX Data Types
  - MicroVAX Subset of VAX Instruction Set
  - Full VAX Memory Management

- Floating Point Processor Chip (on KA630-AA and KA630-AC only)
  - Subset of VAX Floating Point Data Types
  - Subset of VAX Floating Point Instruction Set

- Local Memory
  - 256KB or 1MB of on-board local memory
  - Supports up to two MS630 Memory Expansion Modules, each containing 1MB, 2MB, 4MB or 8MB of additional local memory
  - Byte parity generation and checking for all local memory

- 64KB Boot and Diagnostic ROM
  - Subset VAX Console Program
  - Power-up Diagnostics
  - Boot Programs for Standard Devices

- Console Serial Line Unit
  - Accessed via four VAX Internal Processor Registers
  - Externally Settable Baud Rates

- KA630 Interprocessor Communication Register

- Interval Timer
  - 10 Millisecond Interrupts
  - Interrupts Enabled via Internal Processor Register

- Time of Year Clock

- Q22-Bus Interface
  - Direct Memory Accessing (DMA)
  - Q22-Bus Map which allows DMA to access Local Memory via a 4MB Window divided into 8192 independent pages
  - Arbiter KA630 Fields Q22-Bus Interrupt Requests BR7-4
  - 240 Ohm Termination

3.0 Physical Specifications

3.1 Dimensions

The MS630-AA is a dual height module:

Height	-	5.187 +.015/ -.020 inches
Length	-	8.430 +.010/ -.010 inches
Width	-	.375 inches maximum (non-conductive) .343 inches maximum (conductive)

The KA630-A, MS630-BA, MS630-BB and MS630-CA are quad height modules:

Height	-	10.457 +.015/ -.020 inches
Length	-	8.430 +.010/ -.010 inches
Width	-	.375 inches maximum (non-conductive) .343 inches maximum (conductive)

Note: Width, as defined for digital equipment modules, is the height of components above the surface of the module.

```
+------------------------------------------------------+
!    ------------               ------     ----        !
!  !      J1      !           ! J2   !   ! J3 !        !
!    ------------               ------     ----        !
!                                                      !
!                                                      !
!                                                      !
!                                                      !
!                                                      !
!                                                      !
!                                                      !
!                                                      !
!                                                      !
+--+ Row D  +---+ Row C  +--+ Row B  +---+ Row A  +--+
   !________!   !________!  !________!   !________!
```

Front View of KA630 Connector Locations
Figure 3-1

3.2 KA630 Module Pinouts

The KA630 AB row module pinouts are compatible with the Q22-Bus specification (DEC standard 160). The SRUN L signal appears on pin AF1. The CD row module pinouts utilize the CD-Interconnect to communicate with up to two memory expansion modules.

Note: The KA630 Module can not be used in slots for which the Q22-Bus is connected to both the AB and CD rows (Q22/Q22 configuration). The backplane CD rows must be compatible with the entire CD-Interconnect specification (required for the use of MS630 modules) or must make no connections except for the +5 volt and ground pins designated by the CD-Interconnect specification.

Appendix A summarizes the module pinouts for the KA630 module.

3.3 KA630 CPU Module Connector Pinouts

Figure 3-1 shows the locations of the KA630 Memory Expansion Connector (J1), the Configuration and Display Connector (J2) and the Console SLU Connector (J3).

3.3.1 KA630 Memory Expansion Connector Pinouts

The Memory Expansion Connector (J1) is a 50-pin connector which features the following control, data and ground signals:

BUFEN 1:0	L	(2 pins)
BDIRT	L	(1 pin)
PE 3:0	H	(4 pins)
MD 31:00	H	(32 pins)
GND		(11 pins)

The CD interconnect (refer to section 3.2 and Appendix A) features the following 29 additional address and control signals for the memory expansion modules:

MAA	09:00	H	(10 pins)
RAS	7:0	H	(8 pins)
BMCAS	3:0	H	(4 pins)
BMSWT	2:1	H	(2 pins)
MSID	4:0	L	(5 pins)

3.3.2 Configuration and Display Connector Pinouts

The Configuration and Display Connector (J2) is a 20-pin connector which features the following pinouts:

Pin	Mnemonic	Meaning
01	GND	Ground.
02	GND	
03	GND	
04	CPU CD0 L	CPU Code <1:0>. This 2-bit code determines whether the KA630 is configured as the arbiter or as one of the three auxiliaries:
05	CPU CD1 L	

CPU CD <1:0>	Configuration
00	KA630 Arbiter
01	KA630 Auxiliary #1
10	KA630 Auxiliary #2
11	KA630 Auxiliary #3

CPU Code <1:0> can be read by software via the Boot and Diagnostic Register (section 10.1).

In the Mayflower system, these signals are not connected at the FCC Cutout, but are negated by the KA630 pull-up resistors.

Pin	Mnemonic	Meaning
06	GND	Ground.
07	DSPL 00 L	Display Register Bits 03:00. When asserted, each of these four output signals lights a corresponding external LED. DSPL <03:00> are asserted (low) by power up and by the negation of DCOK. They are updated by the boot and diagnostic programs via the Boot and Diagnostic Register. Writing a "1" asserts the corresponding signal; writing a "0" negates it.
08	DSPL 01 L	
09	DSPL 02 L	
11	DSPL 03 L	
10	BTRY VCC	Battery Backup Voltage for TOY Clock.
12	GND	Ground.
13	BDG CD0 L	Boot and Diagnostic Code <1:0>. This 2-bit code can be read by software via the Boot and Diagnostic Register (section 10.1). The KA630 ROM program may use BDG CD <1:0> to select various Boot Device or Diagnostic test parameters at power up and at system restart. In the Mayflower system, BDG CD <1:0> is provided by a 3-position switch on the CK-KA630-A (the manufacturing test code, "3", can not be selected).
14	BDG CD1 L	

Pin	Mnemonic	Meaning
15	HLT ENB L	Halt Enable. This input signal controls the response to the halt conditions. If HLT ENB is asserted (low), then the KA630 halts and enters the console program if: 1. Program executes a Halt instruction in Kernel Mode 2. Console detects a break character 3. Q22-Bus Halt line is asserted (but only if the KA630 is configured as an arbiter CPU) 4. The Interprocessor Communication Register AUX HLT bit is set (but only if the KA630 is configured as an auxiliary CPU) If HLT ENB is negated, then the Halt line and break character are ignored and the ROM program responds to a halt instruction by restarting or rebooting the system. If HLT ENB is negated, and if the KA630 is configured as an auxiliary, the ROM program responds to assertion of the ICR AUX HLT bit by rebooting. HLT ENB can be read by software via the Boot and Diagnostic Register (section 10.1). In the Mayflower system, HLT ENB originates from a switch on the FCC Cutout.
16	GND	Ground.
17 18 19	BRS 00 L BRS 01 L BRS 02 L	Baud Rate Select <02:00>. This 3-bit code selects the console terminal baud rate. In the Mayflower system, BRS <02:00> is provided by an 8-position switch on the FCC Cutout.
20	+5V	Fused +5 Volts

Note: The KA630 module provides 10K pull-up resistors for the eight input signals (pins 4-5, 13-15 and 17-19).

3.3.3 Console SLU Connector Pinouts

The Console SLU Connector (J3) is a 20-pin connector which features the following pinouts:

Pin	Mnemonic	Meaning
01	-	Reserved. This signal is asserted (+12 volts) whenever the on-board initialize signal is negated. Use of this signal for Data Set Ready is not recommended as it may not be implemented in future designs (refer to +12 volts, pin 10 below).
02	GND	Ground.
03	SLU OUT L	Console SLU Output from the KA630 module.
04	GND	Ground.
05	GND	Ground.
06		Key (No Pin).
07	SLU IN +	Console SLU Differential Inputs to the KA630 module. The received serial data connects to SLU IN -. The signal return connects to SLU +.
08	SLU IN -	
09	GND	Ground.
10	+12V	Fused +12V. The Mayflower System uses +12V for the data set ready signal.

3.4 MS630 Module Pinouts

The MS630-AA memory module is a dual height module which mounts in the CD rows of the next successive slot after either a KA630 module or another MS630 module.

The MS630-BA, MS630-BB and MS630-CA memory modules are quad height modules, each of which can mount in the next successive slot after either a KA630 module or another MS630 module.

The MS630 AB row module pinouts connect with +5 volts (pins AA2, BA2 and BV1) and ground (pins AC2, AJ1, AM1 AT1, BC2, BJ1, BM1 and BT1) only. The MS630 also connects pin AM2 to pin AN2 (passing BIAK) and pin AR2 to AS2 (passing BDMG). The CD row module pinouts require the CD-Interconnect to communicate with the KA630 module and/or another MS630 module.

Note: The MS630 Modules can not be used in slots for which the Q22-Bus is connected to both the AB and CD rows (Q22/Q22 configuration). The backplane CD rows must be compatible with the CD-Interconnect specification.

Appendix A summarizes the module pinouts for the MS630 modules.

```
+------------------------------------------------------+
!   ------------                                        !
!  !     J1     !                                       !
!   ------------                                        !
!                                                       !
!                                                       !
!                                                       !
!                                                       !
!                                                       !
!                                                       !
!                                                       !
!                                                       !
!                                                       !
+-+  Row D  +---+  Row C  +--+  Row B  +---+  Row A  +-+
  !_________!   !_________!  !_________!   !_________!
```

Front View of the MS630 Connector Location
Figure 3-2

3.5 MS630 Connector Pinouts

The Memory Expansion Connector (J1) is a 50-pin connector which features the following control, data and ground signals:

BUFEN 1:0	L	(2 pins)
BDIRT	L	(1 pin)
PE 3:0	L	(4 pins)
MD 31:00	H	(32 pins)
GND		(11 pins)

Refer to the MS630 MicroVAX II Memory Module Specification for additional details on the memory interconnects.

4.0 Electrical Specifications

4.1 DC Power Consumption

The KA630-A CPU module power requirements are:

	+5V +/- 5%	+12V +/- 5%	
KA630-AA (1MB, FP)	6.2	0.14	Amps maximum
KA630-AB (1MB, no FP)	5.9	0.14	
KA630-AC (256KB, FP)	TBD	0.14	
KA630-AD (256KB, no FP)	TBD	0.14	

Note: Typical currents are 10% less than the specified maximum.

The MS630-A/B module power requirements are determined for the refresh-only, (non-read/non-write) condition. Reading or writing memory on one of these modules significantly increases the required power, but results in a corresponding decrease in the power required by the memory on the CPU module. For memory module power requirements under full operating conditions, refer to the MS630 Memory Module Specification.

The MS630-A/B module power requirements under refresh-only conditions are:

	+5V +/- 5%	
MS630-AA (1MB)	1.0	Amps maximum
MS630-BA (2MB)	1.3	
KA630-BB (4MB)	1.8	
KA630-CA (8MB)	TBD	

4.2 Bus Loads

The KA630-A CPU Bus Loads are:

2.7	AC Loads
1.0	DC Loads

The MS630 modules do not place any DC or AC loads on the Q-Bus.

4.3 DCOK Signal

The KA630 receives the BDCOK signal from the Q22-Bus and uses it to initialize the MicroVAX Chip and all clearable KA630 registers. The KA630 will perform a system reboot if DCOK is negated (for 100 nsec or longer) and then reasserted.

Note: The arbiter KA630 asserts BINIT L 200 nsec maximum after the negation of BDCOK and maintains BINIT L assertion until 2 msec min after assertion of BDCOK.

4.4 POK Signal

The KA630 receives the BPOK signal from the Q22-Bus and uses it for Power Up/Power Down sequencing. During Power Up, the KA630 control logic suspends instruction execution by asserting the MicroVAX chip DMR input. After POK has been asserted, this DMR input is negated and the MicroVAX chip begins execution of the KA630 console program. When POK is negated, the KA630 traps through the Power Down vector location.

4.5 Battery Backup Specifications

When DC power is supplied to the KA630 module, it charges the external batteries from +5 volts through a 220 ohm resistor.

When DC power is removed from the KA630 module, it drains the external batteries at a rate of 200 uamps maximum (50 uamps measured typical).

Note: These batteries supply power to the KA630 Time of Year Clock only. There are no battery backup hooks for the memory system.

5.0 Environmental and MTBF Specifications

5.1 Storage Conditions

The KA630 module has an ambient storage temperature range of -40'C (-40'F) to +65'C (149'F). Storage relative humidity is 10% to 90%, non-condensing, altitudes to 9.1 km (50,000 ft).

5.2 Operating Conditions

The KA630 module meets or exceeds the requirements for operation within a system placed in a DEC Standard 102 Class B Environment.

The operating temperature for a KA630 module mounted in a box within a cabinet is 5'C (41'F) to 50'C (122'F) ambient at the module. The maximum outlet temperature rise allowed is 5'C (9'F) above 40'C (104'F). Derate maximum temperature by 1'C for each 1000 meters (1'F for each 1000 ft) of altitude. Operating relative humidity is 10% to 90%, non-condensing.

The airflow required to meet these specifications is 250 lfm.

Note: The module will operate with a 150 lfm airflow if the maximum temperature is restricted to 40'C (104'F).

Note: When the next revision of MicroVAX chip is available, the maximum operating temperature should be increased to allow for a class C operating environment.

5.3 Mean Time Before Failure -- MTBF (Estimate)

The estimated hard error rates for the KA630 and MS630 modules are as follows:

	Class B Environment	Class C Environment
KA630-AA	25.0 khrs.	18.5 khrs.
KA630-AB	25.0	18.5
KA630-AC	25.0	18.5
KA630-AD	25.0	18.5
MS630-AA	93.0	66.0
MS630-BA	59.0	42.0
MS630-BB	39.0	27.0

Calculations of the soft error rates are based on the following assumptions:

1. Although all 256KB RAM chips are currently spec'ed to have an active soft error rate of 10 errors per chip per million hours, certain RAM's (sold by NEC, Fujitsu and Toshiba) have demonstrated a soft error rate of less than 3 errors per chip per million hours when the chips are continuously accessed. Memory Engineering is negotiating with these vendors to bring their specified soft error rate in line with test data.

2. Based on an understanding of how alpha particles cause soft error rates, it can be stated that, for a RAM chip which is not being accessed, the probability of a soft error is reduced by a factor of 3.

3. During normal system operation, a substantial number of soft errors will never be detected because the location in which they occur will be overwritten before being read.

Based on assumptions one and two above, the soft error rate for MicroVAX II systems is specified at 108 soft errors (3 errors times 36 RAM chips) per million hours for the first megabyte of memory and 36 soft errors per million hours for each additional megabyte of memory. Assumption three provides assurance that the observed soft error rate will be less that the specified rate, even if the per chip soft error rates are somewhat greater than assumed above.

The estimated soft error rates for the KA630 and MS630 modules are as follows:

Memory Size	Soft Error MTBF	Memory Size	Soft Error MTBF
1 MB	9,259 hours	6 MB	3,472 hours
2 MB	6,944	7 MB	3,086
3 MB	5,556	8 MB	2,778
4 MB	4,630	9 MB	2,525
5 MB	3,968		

Note: The soft error rate for a single KA630-AC or KA630-AD CPU module (256KB memory) is 27,778 hours. If combined with one or more MS630 modules, round the total memory size up to the next full megabyte and use the table given above.

For a 9 MB system, running 3 shift operation (24 hours a day) for 7 days each week, the specified soft error rate is one error every 15 weeks.

6.0 Central Processor

This section provides summary information about the MicroVAX CPU chip and the MicroVAX architecture. It is not intended as a complete reference, but rather to give an overview of the user-visible features. For a complete description, consult the "MicroVAX CPU Chip Engineering Specification" and the "MicroVAX Architecture Reference Manual".

6.1 MicroVAX CPU Chip Description

The Central Processor and Memory Management functionality is contained within the 68-pin MicroVAX CPU chip. This MicroVAX chip is a 32-bit virtual memory microprocessor, implemented in ZMOS (double metal NMOS). Its key features are:

1. Subset VAX data types. The MicroVAX CPU chip supports the following subset of the VAX data types: byte, word, longword, quadword, character string, and variable length bit field. Support for f_floating, d_floating, and g_floating is available via an external floating point unit. Support for the remaining VAX data types can be provided via macrocode emulation.

2. Subset VAX instruction set. The MicroVAX CPU chip implements the following subset of the VAX instruction set: integer and logical, address, variable length bit field, control, procedure call, miscellaneous, queue, MOVC3/MOVC5, and operating system support. Floating point is implemented via an optional external floating point unit. The remaining VAX instructions can be implemented via macrocode emulation (the MicroVAX chip provides microcode assists for the emulation of the character string, decimal string, EDITPC and CRC instructions).

   Note: If there is no optional floating point unit, the floating point instructions may also be emulated via macrocode emulation.

3. Full VAX memory management. The MicroVAX CPU chip includes a demand paged memory management unit which is fully compatible with VAX memory management. System space addresses are virtually mapped through single level page tables, process space addresses through double level page tables.

4. Industry standard external interface. The MicroVAX CPU chip's external interface is a 32-bit extension of the industry standard microprocessor interface. The MicroVAX CPU chip can be easily interfaced to industry peripheral chips from Motorola, National, and other vendors.

5. Large virtual and physical address space. The MicroVAX CPU chip supports four gigabytes (2**32) of virtual memory, and one gigabyte (2**30) of physical memory.

6. High performance. At its maximum frequency, the MicroVAX CPU chip achieves a 200 nsec microcycle and a 400 nsec I/O (memory) cycle.

7. Single package. The MicroVAX CPU chip is packaged in a standard 68-pin surface mounted chip carrier and requires no special clock generator or support chips.

6.2 Processor State

The processor state consists of that portion of a process's state which is stored in processor registers rather than in memory. This section describes the general purpose register set, the Processor Status Longword and the Processor Registers which are accessed via the Move To Processor Register (MTPR) and Move From Processor Register (MFPR) instructions.

Non-privileged software can access the general purpose register set and bits 15:00 of the Processor Status Longword (i.e. the Processor Status Word). The Processor Registers and bits 31:16 of the Processor Status Longword can only be accessed by privileged software.

6.2.1 General Purpose Registers

There are 16 general purpose registers denoted Rn where n is in the range 0 through 15. The bits of a register are numbered from the right 0 through 31:

```
 3
 1                                                           0
+-------------------------------------------------------------+
!                                                             !
+-------------------------------------------------------------+
```

Certain of these registers have been assigned special meaning by the VAX-11 architecture:

1. R15 is the program counter (PC). The PC contains the address of the next instruction byte of the program.

2. R14 is the stack pointer (SP). The SP contains the address of the top of the processor defined stack.

3. R13 is the current frame pointer (FP). The VAX-11 procedure call convention builds a data structure on the stack called a stack frame. The FP contains the address of the base of this data structure.

4. R12 is the argument pointer (AP). The VAX-11 procedure call convention uses a data structure termed an argument list. The AP contains the address of the base of this data structure.

6.2.2 Processor Status Longword

The Processor Status Longword is implemented per the MicroVAX Architectural Reference Manual which may be referenced for further information on these bits.

```
 3 3 2 2 2 2 2 2 2 2 2 2        1 1
 1 0 9 8 7 6 5 4 3 2 1 0        6 5               8 7 6 5 4 3 2 1 0
+-+-+---+-+-+---+---+-+---------+---------------+-+-+-+-+-+-+-+-+-+
| | |   |F| |   |   |M|         |               | | | | | | | | | |
|C|T|   |P|I|CUR|PRV|B|         |               |D|F|I| | | | | | |
|M|P|MBZ|D|S|MOD|MOD|Z|   IPL   |      MBZ      |V|U|V|T|N|Z|V|C|
+-+-+---+-+-+---+---+-+---------+---------------+-+-+-+-+-+-+-+-+-+
```

Bit(s)	Mnemonic	Meaning
31	CM	Compatibility Mode. This bit always reads as zero, loading a "1" into this bit causes a reserved operand trap. Note: The Compatibility Mode Instructions can be emulated by macrocode. Since the emulation software runs in native mode, the CM bit is never actually set.
30	TP	Trace Pending.
29:28	-	Must be zero.
27	FPD	First Part Done.
26	IS	Interrupt Stack.
25:24	CUR	Current Mode.
23:22	PRV	Previous Mode.
21	-	Must be zero.
20:16	IPL	Interrupt Priority Level.

Bit(s)	Mnemonic	Meaning
15:08	-	Must be zero.
07	DV	Decimal Overflow Trap Enable. This read/write bit has no effect on MicroVAX hardware; it can be used by macrocode which emulates VAX decimal instructions.
06	FU	Floating Underflow Fault Enable.
05	IV	Integer Overflow Trap Enable.
04	T	Trace Trap Enable.
03	N	Negative Condition Code.
02	Z	Zero Condition Code.
01	V	Overflow Condition Code.
00	C	Carry Condition Code.

6.2.3 Processor Registers

The Processor Registers can be accessed through the MFPR and MTPR privileged instructions.

6.2.3.1 Processor Register Summary

Each of the Processor Registers listed in the table below falls into one of the following categories:

1	=	implemented by MicroVAX Chip as specified in MicroVAX SRM
2	=	implemented external to the MicroVAX, by the KA630 logic
3	=	read as zero, nop on write
4	=	implemented by MicroVAX Chip uniquely
5	=	access not allowed; results in reserved operand fault

An "R" following the category number indicates that the register is cleared by power up and by the negation of DCOK.

Number	Register Name	Mneumonic	Type	Category
0	Kernel Stack Pointer	KSP	r/w	1
1	Executive Stack Pointer	ESP	r/w	1
2	Supervisor Stack Pointer	SSP	r/w	1
3	User Stack Pointer	USP	r/w	1
4	Interrupt Stack Pointer	ISP	r/w	1
5	reserved			5
6	reserved			5
7	reserved			5
8	P0 Base Register	P0BR	r/w	1
9	P0 Length Register	P0LR	r/w	1
10	P1 Base Register	P1BR	r/w	1
11	P1 Length Register	P1LR	r/w	1
12	System Base Register	SBR	r/w	1
13	System Length Register	SLR	r/w	1
14	reserved			5
15	reserved			5
16	Process Control Block Base	PCBB	r/w	1
17	System Control Block Base	SCBB	r/w	1
18	Interrupt Priority Level	IPL	r/w	1R
19	AST Level	ASTLVL	r/w	1R
20	Software Interrupt Request	SIRR	w	1
21	Software Interrupt Summary	SISR	r/w	1R
22	Interprocessor Interrupt	IPIR	r/w	5
23	CMI Error Register	CMIERR	r/w	5
24	Interval Clock Control/Status	ICCS	r/w	4R
25	Next Interval Count	NICR	w	3
26	Interval Count	ICR	r	3
27	Time Of Year	TODR	r/w	3
28	Console Storage Receiver Status	CSRS	r/w	3
29	Console Storage Receiver Data	CSRD	r	3
30	Console Storage Transmit Status	CSTS	r/w	3
31	Console Storage Transmit Data	CSTD	w	3
32	Console Receiver C/S	RXCS	r/w	2R
33	Console Receiver D/B	RXDB	r	2R
34	Console Transmit C/S	TXCS	r/w	2R
35	Console Transmit D/B	TXDB	w	2R
36	Translation Buffer Disable	TBDR	r/w	3
37	Cache Disable	CADR	r/w	3
38	Machine Check Error Summary	MCESR	r/w	3
39	Cache Error	CAER	r/w	3
40	Accelerator Control/Status	ACCS	r/w	5
41	Console Saved ISP	SAVISP	r/w	4
42	Console Saved PC	SAVPC	r/w	4
43	Console Saved PSL	SAVPSL	r/w	4
44	WCS Address	WCSA	r/w	5
45	WCS Data	WCSB	r/w	5
46	reserved			5
47	reserved			5

Number	Register Name	Mneumonic	Type	Category
48	SBI Fault/Status	SBIFS	r/w	3
49	SBI Silo	SBIS	r	3
50	SBI Silo Comparator	SBISC	r/w	3
51	SBI Maintenance	SBIMT	r/w	3
52	SBI Error Register	SBIER	r/w	3
53	SBI Timeout Address Register	SBITA	r	3
54	SBI Quadword Clear	SBIQC	w	3
55	IO Bus Reset	IORESET	w	2
56	Memory Management Enable	MAPEN	r/w	1R
57	TB Invalidate All	TBIA	w	1
58	TB Invalidate Single	TBIS	w	1
59	TB Data	TBDATA	r/w	3
60	Microprogram Break	MBRK	r/w	3
61	Performance Monitor Enable	PMR	r/w	3
62	System Identification	SID	r	1
63	Translation Buffer Check	TBCHK	w	1
64:127	reserved			5

6.2.3.2 Category One Processor Registers

Processor registers which are implemented as specified by the MicroVAX or VAX Architecture Reference Manual are classified as category one processor registers. Section 6.2.3.1 lists all category one registers implemented by the MicroVAX chip.

The following category one registers are also referenced in other sections of this specification:

Number	Register Name	Mnemonic	Section
8	P0 Base Register	P0BR	7.4.1
9	P0 Length Register	P0LR	7.4.1
10	P1 Base Register	P1BR	7.4.2
11	P1 Length Register	P1LR	7.4.2
12	System Base Register	SBR	7.3
13	System Length Register	SLR	7.3
17	System Control Block Base	SCBB	6.4.5
18	Interrupt Priority Level	IPL	6.4.1
20	Software Interrupt Request	SIRR	6.4.1
21	Software Interrupt Summary	SISR	6.4.1
56	Memory Management Enable	MAPEN	7.2
57	TB Invalidate All	TBIA	7.2
58	TB Invalidate Single	TBIS	7.2
62	System Identification	SID	6.7

6.2.3.3 Category Two Processor Registers

Those KA630 processor registers which are implemented external to the MicroVAX chip are classified as category two processor registers. Refer to the following sections for a description of these registers:

Number	Register Name	Mnemonic	Section
32	Console Receiver C/S	RXCS	13.2.1
33	Console Receiver D/B	RXDB	13.2.2
34	Console Transmit C/S	TXCS	13.2.3
35	Console Transmit D/B	TXDB	13.2.4
55	IO Bus Reset	IORESET	14.1

6.2.3.4 Category Four Processor Registers

Processor registers which are implemented within the MicroVAX chip and which are unique to the MicroVAX chip (i.e. they are not contained in the MicroVAX or VAX Architecture Reference Manual). Refer to the following sections for a description of these registers:

Number	Register Name	Mnemonic	Section
24	Interval Clock Control/Status	ICCS	12.1
41	Console Saved ISP	SAVISP	6.4.4
42	Console Saved PC	SAVPC	6.4.4
43	Console Saved PSL	SAVPSL	6.4.4

6.3 Instruction Set

The MicroVAX Chip implements all instructions in the following VAX instruction groups:

1. Integer arithmetic and logical
2. Address
3. Variable length bit field
4. Control
5. Procedure call
6. Miscellaneous
7. Queue
8. Character string moves (MOVC3 and MOVC5)

The following instruction groups are implemented by the optional MicroVAX floating Point Chip (if the optional chip is not present, then they may be emulated in software):

1. F_floating
2. G_floating
3. D_floating

The MicroVAX chip provides special microcode hooks to aid the emulation of the following instruction groups by macrocode:

1. Character string (except MOVC3 and MOVC5)
2. Decimal string
3. CRC
4. Edit

The following instruction groups are not implemented, but may be emulated by macrocode:

1. H_floating
2. Octaword
3. Compatibility Mode Instructions

Appendix F lists the entire VAX instruction set, indicating which instructions are implemented in the optional floating point hardware, which instructions are emulated and which instructions are not implemented.

6.4 Exceptions And Interrupts

Both exceptions and interrupts divert execution from the normal flow of control. An exception is typically handled by the current process (e.g. an arithmetic overflow) while an interrupt is typically transfers control outside the process (e.g. an interrupt from an external hardware device).

6.4.1 Interrupts

The MicroVAX architecture has 31 interrupt levels which are used as follows:

Interrupt Levels	Interrupt Condition
non-maskable	HALT L asserted
1F	unused
1E	PWRFL L asserted
19 - 1D	unused
18	unused
17	BR7 L asserted
16	Interval Timer Interrupt, BR6 L asserted
15	BR5 L asserted
14	Console Terminal Interrupts, Interprocessor Doorbell Interrupts, BR4 L asserted
10 - 13	unused
01 - 0F	software interrupt request

Note: Because the Q22-Bus does not allow differentiation between the four bus grants levels (i.e a BR7 device could grab a level 4 bus grant), the KA630 CPU must set the IPL = 17 after responding to any interrupt request BR7-4. The IPL is set = 14 after a console terminal or interprocessor doorbell interrupt, and it is set = 16 after an interval timer interrupt.

Note: When the KA630 is configured as an auxiliary CPU it ignores Q22-Bus BR7-4 interrupt requests, but does respond to IPL 14 requests from its own console serial line unit and from its interprocessor doorbell (in that order of priority). It also responds to interrupt requests from its own interval timer.

The interrupt system is controlled by the Interrupt Priority Level Register (IPL, corresponds to PSL<20:16>), the Software Interrupt Request Register (SIRR), and the Software Interrupt Summary Register (SISR).

```
 3
 1                                                   5 4         0
+---------------------------------------------------+-----------+
|                ignored, returns 0                 |PSL<20:16>|  :IPL
+---------------------------------------------------+-----------+

 3
 1                                                      4 3      0
+------------------------------------------------------+--------+
|                      ignored                         |request|  :SIRR
+------------------------------------------------------+--------+
 3                           1 1
 1                           6 5                             0
+----------------------------+-----------------------------+--+
|                            | Pending Software Interrupts |M|
|                            |                             |B|  :SISR
|                            |F E D C B A 9 8 7 6 5 4 3 2 1|Z|
+----------------------------+-----------------------------+--+
```

6.4.2 Exceptions

The MicroVAX architecture recognizes six classes of exceptions.

exception class	instances
arithmetic traps/faults	integer overflow trap
	integer divide by zero trap
	subscript range trap
	floating overflow fault
	floating divide by zero fault
	floating underflow fault
memory management exceptions	access control violation fault
	translation not valid fault
operand reference exceptions	reserved addressing mode fault
	reserved operand fault or abort
instruction execution exceptions	reserved/privileged instr. fault
	emulated instruction faults
	extended function fault
	breakpoint fault
tracing exception	trace trap
system failure exceptions	memory read error abort
	memory write error abort
	kernel stack not valid abort
	interrupt stack not valid abort
	machine check abort

6.4.3 Machine Check Parameters

In response to a machine check, the following parameters are pushed on the stack:

```
+-------------------------------------------------------+
|              byte count (0000000C hex)                |  :SP
+-------------------------------------------------------+
|              machine check code                       |
+-------------------------------------------------------+
|              most recent virtual address              |
+-------------------------------------------------------+
|              internal state information               |
+-------------------------------------------------------+
|                      PC                               |
+-------------------------------------------------------+
|                      PSL                              |
+-------------------------------------------------------+
```

The parameters are:

machine check code (hex):

1	=	impossible microcode state (FSD)
2	=	impossible microcode state (SSD)
3	=	undefined FPU error code 0
4	=	undefined FPU error code 7
5	=	undefined memory management status (TB miss)
6	=	undefined memory management status (M = 0)
7	=	process PTE address in P0 space
8	=	process PTE address in P1 space
9	=	undefined interrupt ID code
80	=	read bus error, VAP is virtual address
81	=	read bus error, VAP is physical address
82	=	write bus error, VAP is virtual address
83	=	write bus error, VAP is physical address

most recent virtual address:

<31:0>	=	current contents of VAP register

internal state information:

<28:24>	=	current contents of ATDL register
<23:20>	=	current contents of STATE<3:0>
<19:16>	=	current contents of ALU cond codes
<14>	=	current contents of VAX restart bit
<7:0>	=	PC increment at the time of the exception (reported as zero if FPD set in saved PSL)

PC:	<31:0>	=	PC at the start of the current instructions
PSL:	<31:0>	=	current contents of PSL

6.4.4 Halt Conditions

If the hardware or kernel software environment becomes severely corrupted, the chip may be unable to continue normal processing. In these instances, the chip passes control to recovery code beginning at physical address 20040000 (hex). The previous state of the machine is stored in temporary registers which may be read as Processor Registers via the MFPR instruction:

1. IPR 42 contains the saved PC
2. IPR 43 contains the saved PSL, the saved Map En bit and the error code
    a. IPR console.psl bits<31:16,07:00> contain the saved PSL
    b. IPR console.psl bit<15> contains the saved Map En bit
    c. IPR console.psl bits<14:08> contain the error code
3. IPR 41 contains the previous interrupt stack pointer

Note: There are severe restrictions on the usage of these saved values (e.g. they must be accessed before executing certain instructions which use the registers for temporary storage).

The halt process sets the state of the chip as follows:

```
PSL      =    041F0000 (hex)
PC       =    20040000 (hex)
MAPEN    =    0
ASTLVL   =    Unchanged  (set to 4 by power up)
SISR     =    Unchanged  (cleared by power up)
```

The error codes indicating the reason for the halt are as follows:

Error Code	condition
2	assertion of external halt
3	initial power on
4	interrupt stack not valid during exception
5	machine check during machine check or kernel stack not valid exception
6	HALT instruction executed in kernel mode
7	SCB vector bits<1:0> = 11
8	SCB vector bits<1:0> = 10
A	CHMx executed while on interrupt stack
10	ACV or TNV during machine check exception
11	ACV or TNV during kernel stack not valid exception

6.4.5 System Control Block (SCB)

The System Control Block (SCB) consists of two pages which contain the vectors for servicing interrupts and exceptions. The SCB is pointed to by the System Control Block Base Register (SCBB).

```
 3 3 2
 1 0 9                                     9 8              0
+---+---------------------------------------+----------------+
|MBZ|   physical longword address of PCB    |      MBZ       | :SCBB
+---+--------------------------------------------------------+
```

The System Control Block format:

vector	name	type	#param	notes
00	unused	-	-	-
04	machine check	abort	4	refer to section 6.4.3.
08	kernel stack not valid	abort	0	serviced on interrupt stack, IPL is raised to 1F
0C	power fail	interrupt	0	IPL is raised to 1E
10	reserved/privileged instruction	fault	0	
14	extended instruction	fault	0	XFC instruction
18	reserved operand	fault/ abort	0	not always recoverable
1C	reserved addressing mode	fault	0	
20	access control violation	fault	2	parameters are virtual address, status code
24	translation not valid	fault	2	parameters are virtual address, status code
28	trace pending (TP)	fault	0	
2C	breakpoint instruction	fault	0	
30	unused	-	-	compatibility mode in VAX
34	arithmetic	trap/ fault	1	parameter is type code

vector	name	type	#param	notes
38-3C	unused	-	-	-
40	CHMK	trap	1	parameter is operand word
44	CHME	trap	1	parameter is operand word
48	CHMS	trap	1	parameter is operand word
4C	CHMU	trap	1	parameter is operand word
50-5C	unused	-	-	-
60-80	unused	-	-	-
84	software level 1	interrupt	0	
88	software level 2	interrupt	0	ordinarily used for AST delivery
8C	software level 3	interrupt	0	ordinarily used for process scheduling
90-BC	software levels 4-15	interrupt	0	
C0	interval timer	interrupt	0	IPL is 16 (INTTIM L)
C4	unused	-	-	-
C8	emulation start	fault	10	same mode exception, FPD = 0: parameters are opcode, PC, specifiers
CC	emulation continue	fault	0	same mode exception, FPD = 1: no parameters
D0-F4	unused	-	-	-
F8	Console Receive	interrupt	0	IPL is 14
FC	Console Transmit	interrupt	0	IPL is 14
100-1FC	adapter vectors	interrupt	0	Not implemented by the KA630
200-3FC	device vectors	interrupt	0	Correspond to Q22-Bus Vectors 000-1FC; KA630 appends the assertion of bit <09>

6.4.5.1 KA630 Assigned Device Vector

The KA630 uses System Control Block (SCB) device vector 204 (hex) for the interprocessor doorbell interrupt (section 14.3.2).

6.5 Hardware Detected Errors

The KA630 detects certain error conditions during program execution. These conditions, and the resultant actions, are described below.

6.5.1 Non-Existent Memory Errors

If the MicroVAX chip attempts a read or write access to a non-existent location in local memory or I/O Space, the KA630 asserts the error (ERR) signal to the chip.

If the MicroVAX chip attempts a read or write access to the Q22-Bus, a bus timeout error can occur. If BRPLY L is not asserted within 10 microseconds following the assertion of BDIN L or BDOUT L, the KA630 asserts the ERR signal to the chip.

If ERR is asserted, the MicroVAX processor responds as follows:

1. For write accesses and for non-prefetch reads, the MicroVAX processor recognizes a machine check and traps through vector 04.

2. For prefetch operations, the MicroVAX processor aborts the prefetch cycle and performs a non-prefetch read if an instruction fetch is required from that location.

6.5.2 Parity Error Detection

Parity errors can be detected during read operations from the local memory address space, from the Q22-Bus memory address space or from the Q22-Bus I/O Page address space.

Memory System Error Register bit <00> (section 9.4.1) enables parity error detection for all reads from local memory, whether it is accessed through local memory address space or through the Q22-Bus memory address space (via the Q22-Bus Map). MSER <00> has no effect on parity error detection for reads from external Q22-Bus memory or devices.

During read operations from the local memory address space, parity is checked only for those bytes designated by the MicroVAX chip Byte Mask signals, BM <3:0>. Because the MicroVAX chip must receive a stable signal on ERR at least 150 nsec before it requires stable data, performance considerations dictate that any parity error which occurs during reads from local memory address space will not cause an ERR assertion during the cycle for which the parity error was detected. Instead, the KA630 asserts ERR for the next cycle and, if that cycle was a pre-fetch read cycle, for the cycle after that as well.

Note: When a parity error occurs during a local memory access via local memory address space, the MicroVAX is allowed to complete that cycle and may execute an instruction which alters the MicroVAX chip's internal state. However, the MicroVAX recognizes a machine check and traps through vector 04 when it attempts the next external cycle.

/Justification for this incompatibility with past VAX-11 processors rests on two considerations: 1) a performance improvement of 10-15% over a design which would stall all read cycles to allow parity detection within the cycle and 2) an estimated local memory parity error rate of from once every six months (KA630 with two MS630-BB modules, each containing 4MB of 256K RAM; three shift operation) to once every ten years (KA630 with no memory expansion modules; single shift operation)./

During read operations from Q22-Bus space (including the access of local memory through the Q22-Bus Map), a parity error is detected if both BDAL <17> and BDAL <16> are asserted. When a parity error is detected, the KA630 asserts ERR to the MicroVAX chip.

Note: When the processor reads local memory via the Q22-Bus memory space, parity is checked on both bytes of each word accessed, even if the processor only requested a single byte.

If ERR is asserted, the MicroVAX processor responds as follows:

1. For non-prefetch reads, the MicroVAX processor recognizes a machine check and traps through vector 04.

2. For prefetch operations, the MicroVAX processor aborts the prefetch cycle and performs a non-prefetch read if an instruction fetch is required from that location.

6.5.3 Interrupt Vector Timeouts

An interrupt vector timeout occurs when BRPLY L is not asserted within 10 microseconds after an interrupt is acknowledged (BIAK L). The KA630 asserts the Error (ERR) signal to its MicroVAX chip. The processor aborts the interrupt cycle and continues as if the interrupt request had never occurred.

6.5.4 "No Sack" Timeouts

A "No Sack" timeout occurs when BSACK L is not asserted within 10 microseconds after a DMA is granted (BDMG L). The timeout is ignored. The KA630 continues as if the DMA request had never occurred.

6.6 Latency Specifications

6.6.1 Interrupt Latency (Estimates)

Interrupt latency is defined as the time between receiving an interrupt request (BIRQ L) and acknowledging the request (BIAK L). The interrupt latency can be divided into the following components:

1. The length of time the processor runs at an interrupt priority level which masks out the interrupt. This time period is highly software dependent.

2. The length of time the processor takes to execute the last instruction before the interrupt. For instructions which do not access the Q-Bus, this time period is as follows:

a.	Longest Non-Preemptable Instruction	2.4 usec
b.	Preempting an Instruction	5.4 usec

   Each Q-Bus access would add approximately 1.0 usec for word accesses and 1.5 usec for longword accesses. A bus timeout, which could occur on the last Q-Bus access, would add an additional 10-15 usec.

3. The length of time it takes the KA630 to gain Q22-Bus mastership. Because the arbiter KA630 is the highest priority DMA device, this period is equal to the time required for the previous bus master to finish its data transfer(s) and relinquish the bus. Eight block mode transfers would typically require about 5 usec (as currently planned, the KA630 will assert DMR when it needs the bus, limiting a block mode device to no more than eight additional transfers). A non-existent memory timeout would typically require 10-15 usec.

Note: This represents a change from the traditional priority structure where DMA devices have a higher priority than either CPU fetches or interrupts. The justifications for this change are:

1. The KA630 CPU usually runs out of local memory, greatly reducing bus utilization by the CPU.
2. Modern DMA devices are buffered and better able to withstand the increase in DMA latency.
3. The result is a significant improvement in interrupt latency which has degraded with the increased use of buffered DMA devices.

6.6.2 Interrupt Service Time

Interrupt service time is defined as the time between acknowledging the interrupt and fetching the first instruction of the service routine. The KA630 service time is 4.4 usec.

6.6.3 DMA Latency

DMA latency is defined as the time between receiving a DMA Request (BDMR L) and granting the request (BDMG L). This calculation has traditionally been made assuming that the DMA request occurs while the CPU has control of the bus and that there are no conflicting DMA requests. The result of this calculation is, therefore, the CPU induced latency. The DMA latency seen by any device is a combination of the CPU induced latency and the latency induced by other DMA devices.

The CPU induced DMA latency is the time required to complete the longest CPU operation which retains control of the bus. The longest KA630 operation which retains control of the bus is a 32-bit read-lock/write-unlock to non-block-mode memory. Typical CPU induced DMA latencies for the KA630 can be summarized as follows:

Cycle	Latency
32-bit read-lock/write (non-block-mode)	4.2 usec
32-bit read-lock/write (block-mode)	3.2
32-bit read (non-block-mode)	2.0
32-bit read (block-mode)	1.5
32-bit write (non-block-mode)	2.0
32-bit write (block-mode)	1.5
16-bit read-lock/write	2.2
16-bit read	1.0
16-bit write	1.0

Note: Two successive byte writes, a word write followed by a byte write and a byte write followed by a word write are subsets of the 32-bit write (non-block-mode). A single byte write is a subset of the 16-bit write.

When a CPU which is the lowest priority device has relinquished control of the bus, it does not regain control of the bus until all DMA requests have been honored. Thus, two high bandwidth devices could exchange control of the bus, effectively locking out the CPU until one of them has completed its set of transfers.

The arbiter KA630 CPU is the highest priority bus device in a system. After it has relinquished control of the bus, it can regain control of the bus during the next bus arbitration.

Note: System level DMA latency calculations must take into account the fact that the arbiter KA630 can request the bus as the highest priority bus device.

6.7 System Identification

As noted in section 6.2.3, the read-only System Identification Register, Processor Register 62, is implemented by the MicroVAX chip. On the KA630-A, and on all other processors which use the MicroVAX chip, the System Identification Register always reads as "0000 0008".

The KA630-A, as must all processors which use the MicroVAX chip, implements a 32-bit System Identification Extension Register at physical location 2004 0004. This 32-bit register exists within the KA630-A console program ROM (section 10.2).

```
 3              2 2             1 1
 1              4 3             6 5                            0
+-----------------------------------------------------------------+
!     SYSCODE     !     VERSION    !           reserved           !
+-----------------------------------------------------------------+
```

Bits	Mnemonic	Meaning
31:24	SYSCODE	System Code. This field reads as "1" for the KA630-A.
23:16	VERSION	Version number of console program ROM.
15:00		Reserved.

7.2   Memory Management Control Registers

Memory management is controlled by three processor registers: Memory Management Enable (MAPEN), Translation Buffer Invalidate Single (TBIS), and Translation Buffer Invalidate ALL (TBIA).

MAPEN contains one bit:

MAPEN<0> = MME enables memory management.

```
 3
 1                                                    1 0
+-----------------------------------------------------+-+
|                                                     |M|
|                         MBZ                         |M|  :MAPEN
|                                                     |E|
+-----------------------------------------------------+-+
```

TBIS controls translation buffer invalidation. Writing a virtual address into TBIS invalidates any entry which maps that virtual address.

```
 3
 1                                                      0
+-------------------------------------------------------+
|                    Virtual Address                    |  :TBIS
+-------------------------------------------------------+
```

TBIA also controls translation buffer invalidation. Writing a zero into TBIA invalidates the entire translation buffer.

```
 3
 1                                                      0
+-------------------------------------------------------+
|                         MBZ                           |  :TBIA
+-------------------------------------------------------+
```

7.3   System Space Address Translation

A virtual address with bits <31:30> = 2 is an address in the system virtual address space.

System virtual address space is mapped by the System Page Table (SPT), which is defined by the System Base Register (SBR) and the System Length Register (SLR). The SBR contains the physical address of the System Page Table. The SLR contains the size of the SPT in longwords, that is, the number of Page Table Entries. The Page Table Entry addressed by the System Base Register maps the first page of system virtual address space, that is, virtual byte address 80000000 (hex).

```
 3 3 2
 1 0 9                                                     2 1 0
+---+-------------------------------------------------------+---+
|MBZ|          physical longword address of SPT             |MBZ| :SBR
+---+-------------------------------------------------------+---+

 3                 2 2
 1                 2 1                                         0
+-------------------+-------------------------------------------+
|         MBZ       |        length of SPT in longwords         | :SLR
+-------------------+-------------------------------------------+


                   3 3 2
                   1 0 9                  9 8      0
                  +---+--------------------+--------+
SVA:              | 2 |                    |  byte  |
(System Virtual   +---+--------------------+--------+
   Address)           |     extract and    |      |
              3      2|2    check length   |      |
              1      3|2                  2|10    |
             +--------+--------------------+--+   |
             |   0    |                    | 0|   |
             +--------+--------------------+--+   |
                                                  |
                                                  |
                              add                 |
                                                  |
             +-----------------------------+--+   |
SBR:         |  Physical Base Adr of SPT   | 0|   |
             +-----------------------------+--+   |
                                                  |
                            yields                |
                                                  |
             +-----------------------------+--+   |
             |     Physical Adr of PTE     | 0|   |
             +-----------------------------+--+   |
                                                  |
                            fetch                 |
                                                  |
            3 3      2 2                          |
            1 0      1 0                  0       |
           +-+--------+--------------------+      |
PTE:       |1|        |         PFN        |      |
           +-+--------+--------------------+      |
         check access | this access check  |      |
                      | in current mode    |      |
                      |                    |      |
                      |2                   |      |
                      |9                  9|8     V 0
Physical Address      +--------------------+--------+
of Data:              |                    |        |
                      +--------------------+--------+
```

System Virtual to Physical Translation

7.4 Process Space Address Translation

A virtual address with bit <31> = 0 is an address in the process virtual address space. Process space is divided into two equal sized, separately mapped regions. If virtual address bit <30> = 0, the address is in region P0. If virtual address bit <30> = 1, the address is in region P1.

7.4.1 P0 Region Address Translation

The P0 region of the address space is mapped by the P0 Page Table (P0PT), which is defined by the P0 Base Register (P0BR) and the P0 Length Register (P0LR). The P0BR contains the system virtual address of the P0 Page Table. The P0LR contains the size of the P0PT in longwords, that is, the number of Page Table Entries. The Page Table Entry addressed by the P0 Base Register maps the first page of the P0 region of the virtual address space, that is, virtual byte address 0.

```
 3 3 2
 1 0 9                                                     2 1 0
+---+-----------------------------------------------------+---+
| 2 |      system virtual longword address of P0PT        |MBZ| :P0BR
+---+-----------------------------------------------------+---+

 3              2 2
 1              2 1                                           0
+----------------+--------------------------------------------+
|       MBZ      |         length of P0PT in longwords        | :P0LR
+----------------+--------------------------------------------+
                   3 3 2
                   1 0 9                  9 8      0
                  +---+--------------------+--------+
PVA:              | 0 |                    |  byte  |
(Process Virtual  +---+--------------------+--------+
    Address)          |     extract and    |      |
              3      2|2    check length   |      |
              1      3|2                  2|10    |
             +--------+--------------------+--+   |
             |   0    |                    | 0|   |
             +--------+--------------------+--+   |
                                                  |
                                                  |
                              add                 |
                                                  |
             +-----------------------------+--+   |
P0BR:        |  Sys Virt Base Adr of P0PT  | 0|   |
             +-----------------------------+--+   |
                                                  |
                            yields                |
                                                  |
             +-----------------------------+--+   |
             |      Virtual Adr of PTE     | 0|   |
             +-----------------------------+--+   |
                                                  |
                    fetch by system space         |
                      translation algorithm,      |
                      including length and        |
                      kernel mode access checks   |
                                                  |
            3 3      2 2                          |
            1 0      1 0                  0       |
           +-+--------+--------------------+      |
PTE:       |1|        |         PFN        |      |
           +-+--------+--------------------+      |
         check access | this access check  |      |
                      | in current mode    |      |
                      |                    |      |
                      |2                   |      |
                      |9                  9|8     V 0
Physical Address      +--------------------+--------+
of Data:              |                    |        |
                      +--------------------+--------+
```

P0 Virtual to Physical Translation

7.4.2 P1 Region Address Translation

The P1 region of the address space is mapped by the P1 Page Table (P1PT), which is defined by the P1 Base Register (P1BR) and the P1 Length Register (P1LR). Because P1 space grows towards smaller addresses, and because a consistent hardware interpretation of the base and length registers is desirable, P1BR and P1LR describe the portion of P1 space that is NOT accessible. Note that P1LR contains the number of non-existent PTEs. P1BR contains the virtual address of what would be the PTE for the first page of P1, that is, virtual byte address 40000000 (hex). The address in P1BR is not necessarily a valid physical address, but all the addresses of PTEs must be valid physical addresses.

```
 3
 1                                                       2 1 0
+-------------------------------------------------------+---+
|           virtual longword address of P1PT            |MBZ| :P1BR
+-------------------------------------------------------+---+

 3             2 2
 1             2 1                                          0
+--------------+---------------------------------------------+
|     MBZ      |          length of P1PT in longwords        | :P1LR
+--------------+---------------------------------------------+
                   3 3 2
                   1 0 9                  9 8      0
                  +---+--------------------+--------+
PVA:              | 1 |                    |  byte  |
(Process Virtual  +---+--------------------+--------+
   Address)           |    extract and     |      |
              3      2|2   check length    |      |
              1      3|2                  2|10    |
             +--------+--------------------+--+   |
             |    0   |                    | 0|   |
             +--------+--------------------+--+   |
                                                  |
                              add                 |
                                                  |
             +-----------------------------+--+   |
P1BR:        |    Virt Base Adr of P1PT    | 0|   |
             +-----------------------------+--+   |
                                                  |
                            yields                |
                                                  |
             +-----------------------------+--+   |
             |      Virtual Adr of PTE     | 0|   |
             +-----------------------------+--+   |
                                                  |
                    fetch by system space         |
                      translation algorithm,      |
                      including length and        |
                      kernel mode access checks   |
                                                  |
            3 3      2 2                          |
            1 0      1 0                  0       |
           +-+--------+--------------------+      |
PTE:       |1|        |         PFN        |      |
           +-+--------+--------------------+      |
         check access | this access check  |      |
                      | in current mode    |      |
                      |                    |      |
                      |2                   |      |
                      |9                  9|8     V 0
Physical Address      +--------------------+--------+
of Data:              |                    |        |
                      +--------------------+--------+
```

P1 Virtual to Physical Translation

7.5 Page Table Entry

The format of a valid page table entry is:

```
 3 3     2 2 2 2 2 2 2 2
 1 0     7 6 5 4 3 2 1 0                                      0
+-+-------+-+-+---+---+---------------------------------------+
|V| PROT  |M|0|OWN| 0 |                  PFN                  | :PTE
+-+-------+-+-+---+---+---------------------------------------+
```

where:

V	=	Valid Bit (must be set)
PROT	=	Protection Code
M	=	Modify Bit
OWN	=	Owner Bits
PFN	=	Page Frame Number

If Bit <31> (the V bit) is clear, the format of the remaining bits are not examined by the hardware.

8.0 Floating Point Processor Description

The Floating Point Processor option consists of a 68 pin ZMOS (double metal NMOS) Floating Point Unit (FPU) chip.

The key features of the FPU chip are as follows:

1. Subset VAX Data types. The MicroVAX FPU chip supports byte, word, longword, f_floating, d_floating and g_floating data types. The h_floating data type is not supported, but may be implemented by macrocode emulation.

2. Subset VAX Instruction Set. The MicroVAX FPU chip implements all VAX floating point instructions except for the h_floating instructions and the floating point instructions which are implemented by the MicroVAX CPU chip. The h_floating data type is not supported, but may be implemented by macrocode emulation.

9.0 KA630 Memory System

The KA630 Memory System consists of the KA630 local memory as well as the Q22-Bus Map which allows Q22-Bus master devices to access this local memory. The Memory System also includes two registers which are used primarily for diagnostic purposes.

9.1 Memory System Summary

The KA630 supports up to 16MB of local memory. The KA630 CPU typically accesses this memory directly, via physical addresses 00000000 - 00FFFFFF (hex). Any Q22-Bus master device, including the KA630 CPU, can access this memory indirectly through the Q22-Bus Map. The Q22-Bus Map contains 8192 mapping registers, each of which can map a page (512 bytes) of Q22-Bus space into a selected page in local memory. Mapping can be independently enabled and disabled for each page.

The KA630 CPU Module accesses the Q22-Bus memory address space via physical addresses 30000000 - 303FFFFF (hex). It accesses the Q22-Bus I/O space via physical addresses 20000000 - 20001FFF.

9.2 KA630 Local Memory

The KA630-AA and KA630-AB MicroVAX CPU Modules feature 1MB of on-board memory; the KA630-AC and KA630-AD MicroVAX CPU Modules feature 256KB on-board memory, The KA630-A module utilizes the backplane CD Interconnect and a 50-pin connector to communicate with up to two MS630 memory expansion modules, each of which may contain 1MB, 2MB, 4MB or 8MB of additional local memory. A KA630 module with two MS630 memory expansion modules can thus have a maximum of 16MB local memory. All local memory performs byte parity generation and checking.

Note: When a KA630 module has 16MB of expansion memory (i.e two MS630-CA modules), its on-board memory is disabled.

9.3 Mapping Registers

The Q22-Bus Map contains 8192 mapping registers, each of which can map a page (512 bytes) of Q22-Bus space into a selected page of local memory.

9.3.1 Mapping Register Format

Each of the mapping registers is a 32 bit long word with the following format:

```
 3 3                            1 1
 1 0                            5 4                           0
+-+-----------------------------+-----------------------------+
!V!                             !        A23 - A09            !
+-+-----------------------------------------------------------+
```

Bit(s)	Mnemonic	Meaning
31	V	Valid. When a mapping register is selected by a Q22-Bus address, the Valid bit determines whether the Q22-Bus map is enabled for that address. If the Valid bit is set, the map is enabled. If the Valid bit is clear, the map is disabled and the KA630 does not respond to that address.
30:15	-	Unused. These bits always read as zero.
14:00	A23-A09	Address bits A23 - A09. When a mapping register is selected by a Q22-Bus address, and if that register's Valid bit is set, then these fifteen bits are used as local memory address bits 23 thru 09. Q22-Bus address bits 08 thru 00 are used as local memory address bits 08 thru 00.

Note: Each mapping register is located on a long word boundary and must be written using long word instructions. Byte and word instructions will load these registers with undefined data.

9.3.2 Mapping Register Addresses (Hex Addresses 2008 XXXX)

The mapping registers are located within the local register space at physical addresses 2008 8000 - 2008 FFFC. They can only be accessed from the on-board processor.

The physical address of each register was chosen so that register address bits <14:02> are identical to Q22-Bus address bits <21:09> of the page which they map.

Register Address	Q22-Bus Addresses Mapped (Hex)	Q22-Bus Addresses Mapped (Octal)
2008 8000	00 0000 - 00 01FF	00 000 000 - 00 000 777
2008 8004	00 0200 - 00 03FF	00 001 000 - 00 001 777
2008 8008	00 0400 - 00 05FF	00 002 000 - 00 002 777
2008 800C	00 0600 - 00 07FF	00 003 000 - 00 003 777
2008 8010	00 0800 - 00 09FF	00 004 000 - 00 004 777
2008 8014	00 0A00 - 00 0BFF	00 005 000 - 00 005 777
2008 8018	00 0C00 - 00 0DFF	00 006 000 - 00 006 777
2008 801C	00 0E00 - 00 0FFF	00 007 000 - 00 007 777

Register Address	Q22-Bus Addresses Mapped (Hex)	Q22-Bus Addresses Mapped (Octal)
.	.	.
.	.	.
.	.	.
.	.	.
2008 FFF0	3F F800 - 3F F9FF	17 774 000 - 17 774 777
2008 FFF4	3F FA00 - 3F FBFF	17 775 000 - 17 775 777
2008 FFF8	3F FC00 - 3F FDFF	17 776 000 - 17 776 777
2008 FFFC	3F FE00 - 3F FFFF	17 776 000 - 17 777 777

9.3.3 Q22-Bus Map Operation

At power up time, the Q22-Bus mapping registers, including the valid bits, are undefined. External access to local memory is disabled so long as the Interprocessor Communication Register (ICR) LM EAE bit is cleared.

After completion of the ROM diagnostic programs which are part of the KA630 console program, an arbiter KA630 enables the mapping registers to map sufficient local memory space to boot the system and then sets the LM EAE bit. When the operating system gains control, it may either invalidate or reassign various pages as required.

After completion of its ROM diagnostic programs, but before setting the LM EAE bit, the auxiliary KA630 ROM programs clear all mapping register valid bits.

The Q22-Bus Map monitors each Q22-Bus cycle and responds if the following three conditions are met:

1. The Interprocessor Communication Register LM EAE bit is set.

2. The Valid bit of the selected mapping register is set.

3. During read operations, the mapping register must map into existent local memory. (During write operations, the KA630 returns Q22-Bus BRPLY before checking for existent local memory; the response depends only on conditions 1 and 2 above).

The translation from Q22-Bus address to local memory physical address is as follows:

```
                   2
                   1                       9 8      0
                  +-------------------------+--------+
Q22-Bus Address   |                         |  byte  |
                  +-------------------------+--------+
                      extract and           |        |
                      use to select         |        |
                      mapping register      |        |
                                            !        !
                                            !        !
                                            !        !
Selected Mapping Register:                  !        !
                                            !        !
 3 3        1 1                             !        !
 1 0        5 4                            0!        !
+-+----------+------------------------------+        |
!V!          !                              !        !
!-+----------+------------------------------+        !
             !                              !        !
             !                              !        !
             !                              !        !
             !                              !        !
             !2                             V        V
             !3                            9 8      0
             +------------------------------+--------+
             !   Physical Address of Local Memory    !
             +------------------------------+--------+
```

9.4 Memory System Registers

The three registers associated with the Memory System are located in the local register I/O address space and can only be accessed by the on-board processor. Software uses the Memory System Error Register to monitor parity and non-existent memory errors and to control parity generation and checking for the local memory. The CPU Error Address Register contains the address of the page in local memory which caused a parity error during an access by the on-board CPU. The DMA Error Address Register contains the address of the page in local memory which caused a parity error during an access by an external device.

9.4.1 Memory System Error Register (Hex Address: 2008 0004)

The Memory System Error Register (MSER) is located in the local register I/O address space at physical address 2008 0004. It can only be accessed by the on-board processor.

MSER <07:05> and MSER <03> indicate the status of machine check traps through System Control Block (SCB) vector 04. MSER bit <04> is set if an external Q22-Bus device receives a parity error while reading KA630 local memory.

When CPU read operation parity error sets MSER bit <06> or <05>, MSER <09:08> contains a code which identifies the source of the error as Q22-Bus memory, as on-board local memory or as one of the two memory expansion modules. Additional parity errors can not update the contents of MSER <09:08> until after software has cleared bits MSER <06:05>.

MSER bit <01> is a write wrong parity bit which is set for diagnostic purposes only.

MSER bit <00> is a parity error enable bit which enables local memory parity checking for both CPU and DMA reads.

```
 3                                          . 1
 1                                            0 9 8 7 6 5 4 3 2 1 0
+-----------------------------------------------------------------+
!              unused, returns 0               ! ! ! ! ! ! ! !0! ! !
+-----------------------------------------------------------------+
                                                ! ! ! ! ! ! !   ! !
        MEM CD1  -------------------------------+ ! ! ! ! ! !   ! !
        MEM CD0  ---------------------------------+ ! ! ! ! !   ! !
                                                    ! ! ! ! !   ! !
        CPU NXM  -----------------------------------+ ! ! ! !   ! !
        CPU LPE  -------------------------------------+ ! ! !   ! !
        CPU QPE  ---------------------------------------+ ! !   ! !
        DMA QPE  -----------------------------------------+ !   ! !
         MS LEB  -------------------------------------------+   ! !
                                                                ! !
        WRW PAR  -----------------------------------------------+ !
        PAR ENB  -------------------------------------------------+
```

Bit(s)	Mnemonic	Meaning
31:10	-	Unused. Reads as zero.
09 08	MEM CD1 MEM CD0	Memory Code 1 - 0. When one of the two CPU parity error bits (MSER <06:05>) is set, the two read-only MEM CD bits are loaded with a 2-bit code which indicates the source of the parity error per the following table:
		00 - Q22-Bus Memory or Device
		01 - KA630 On-Board Memory
		10 - Memory Expansion Module #1
		11 - Memory Expansion Module #2
		A second parity error will not update this code unless software has cleared the CPU parity error bits. MEM CD <1:0> are cleared by power up, by the negation of DCOK and by writes to the Bus Initialize Register.

Bit(s)	Mnemonic	Meaning
07	CPU NXM	CPU Non-Existent Memory Error. This bit is set by any CPU non-prefetch read or write operation which references non-existent memory, causing a trap through SCB vector 04. Writing a "1" to this bit clears it; writing a "0" to this bit has no effect. CPU NXM is cleared by power up, by the negation of DCOK and by writes to the Bus Initialize Register.
06	CPU LPE	CPU Local Address Space Parity Error. If parity error detection is enabled (MSER <00> is set), then CPU LPE is set by any CPU read access (prefetch or non-prefetch) to local memory address space which causes a parity error. The MicroVAX chip does not receive an error indication on that cycle. The next MicroVAX cycle is aborted, causing a trap through SCB vector 04. Writing a "1" to this bit clears it; writing a "0" to this bit has no effect. CPU LPE is cleared by power up, by the negation of DCOK and by writes to the Bus Initialize Register. Note: Only those memory bytes selected by MicroVAX chip outputs BM <3:0> can cause a CPU LPE parity error. Note: Because the fetch which caused the parity error is not aborted, it could be difficult for software to determine the result of the error. For this reason, parity errors which set this bit are generally treated as fatal errors.
05	CPU QPE	CPU Q22-Bus Address Space Parity Error. CPU QPE is set by any CPU non-prefetch read access to the Q22-Bus Address Space which results in a parity error, causing a CPU trap through SCB vector 04. If the CPU is accessing local memory via the Q22-Bus Map, parity detection is enabled only if MSER <00> is set. If the CPU is accessing the Q22-Bus, parity detection is enabled or disabled at the external Q22-Bus memory or device. Writing a "1" to this bit clears it; writing a "0" to this bit has no effect. CPU QPE is cleared by power up, by the negation of DCOK and by writes to the Bus Initialize Register.

Bit(s)	Mnemonic	Meaning
04	DMA QPE	DMA Q22-Bus Address Space Parity Error. If parity error detection is enabled (MSER <00> is set), then DMA QPE is set by any external read access to KA630 local memory which results in a parity error. This type of parity error does not cause the CPU trap through SCB vector 04. (The DMA device typically interrupts with an error indication). Writing a "1" to this bit clears it; writing a "0" to this bit has no effect. DMA QPE is cleared by power up, by the negation of DCOK and by writes to the Bus Initialize Register.
03	MS LEB	Memory System Lost Error Bit. This bit is set by an operation which sets MSER <06> or <05> after one or both of those bits have already been set. Writing a "1" to this bit clears it; writing a "0" to this bit has no effect. MS LEB is cleared by power up, by the negation of DCOK and by writes to the Bus Initialize Register.
02	-	Unused. Read as zeros.
01	WRW PAR	Write Wrong Parity. If this read-write bit is set, and either the CPU or a DMA device writes to local memory, then wrong parity is written into the parity bits of the MOS RAM's. If this bit is clear, correct parity is written. WWR PAR is cleared by power up, by the negation of DCOK and by writes to the Bus Initialize Register.
00	PAR ENB	Parity Enable. If this read-write bit is set, local memory parity error detection is enabled. If this bit is clear, parity errors are ignored during all CPU and DMA reads from local memory. PAR ENB is cleared by power up, by the negation of DCOK and by writes to the Bus Initialize Register. Note: PAR ENB controls parity detection for all CPU reads from local memory, including accesses via the Q22-Bus Map. PAR ENB has no affect on CPU reads from external Q22-Bus memory.

9.4.2 CPU Error Address Register (Hex Address: 2008 0008)

The CPU Error Address Register (CEAR) is located in the local register I/O address space at physical address 2008 0008. It can only be accessed by the on-board processor and contains valid information only when either MSER <06> (CPU LPE) or MSER <05> (CPU QPE) is set.

The CPU Error Address Register contains the address of the page in local memory which caused a parity error during an access by the on-board CPU. The contents of this register is latched when either MSER <06> or MSER <05> is set. Additional local memory parity errors have no effect on the CEAR until software clears MSER <06:05>.

Local memory address bits <23:09> are loaded into CEAR bits <14:00>. CEAR bits <31:15> always reads as zero.

```
 3                              1 1                             0
 1                              5 4                             
+---------------------------------------------------------------+
!        unused, returns 0        !         Local Memory          !
!                                 !     Address Bits <23:09>      !
+---------------------------------------------------------------+
```

9.4.3 DMA Error Address Register (Hex Address: 2008 000C)

The DMA Error Address Register (DEAR) is located in the local register I/O address space at physical address 2008 000C. It can only be accessed by the on-board processor and contains valid information only when MSER <04> (DMA QPE) is set.

The DMA Error Address Register contains the address of the page in local memory which caused a parity error during an access by an external device. The contents of this register is latched when MSER <04> is set. Additional local memory parity errors have no effect on the DEAR until software clears MSER <04>.

Local memory address bits <23:09> are loaded into DEAR bits <14:00>. DEAR bits <31:15> always reads as zero.

```
 3                              1 1                             0
 1                              5 4                             
+---------------------------------------------------------------+
!        unused, returns 0        !         Local Memory          !
!                                 !     Address Bits <23:09>      !
+---------------------------------------------------------------+
```

9.5 Memory System Operation

The KA630 memory system can perform the following data transfer and memory refresh cycles:

1. MicroVAX accesses local memory directly
2. MicroVAX accesses local memory through Q22-Bus Map
3. MicroVAX accesses on-board registers in the local or Q22-Bus I/O address space.
4. MicroVAX accesses Q22-Bus memory or registers
5. External Q22-Bus device accesses local memory through Q22-Bus Map
6. External Q22-Bus device accesses the KA630 Interprocessor Communication Register (in the Q22-Bus I/O space)
7. KA630 performs memory refresh cycle

Note: The MicroVAX does not require access to the memory system when it accesses internal processor registers (including those implemented external to the MicroVAX chip).

Access to the local memory data/address paths and control of the Q22-Bus are arbitrated independently. An external device can gain control of the Q22-Bus and access a KA630 register (in the Q22-Bus address space) while the MicroVAX chip is accessing local memory. When an external device accesses KA630 local memory, MicroVAX cycles are not interrupted until the Q22-Bus Map has decoded the local memory address.

When the local memory is between cycles, it continually decodes the address on the MicroVAX address/data lines. When it receives a refresh or external device request, it switches off the MicroVAX address/data lines and monitors the address lines from the refresh logic or from the Q22-Bus Map.

When the local memory is between cycles, the arbitrator for the local memory responds to requests in the following order of priority:

1. MicroVAX request
2. External device request
3. Refresh request

When the local memory is completing a cycle, the arbitrator for the local memory responds to requests in the following order of priority:

1. External device request
2. Refresh request
3. MicroVAX request

The local memory cycle time is 400 nsec for all read, write or refresh cycles.

The MicroVAX must have control of the Q22-Bus:

1. Before it can perform any read-lock cycle.
2. Before it can access local memory via the Q22-Bus Map
3. Before it can access the Interprocessor Communication Register or one of the Q22-Bus map registers.
4. Before it can perform Q22-Bus cycles

On an arbiter KA630, which contains the Q22-Bus arbitrator, the MicroVAX has control of the Q22-Bus except when the KA630 has granted an external DMA request. On an auxiliary KA630, the MicroVAX can gain control of the Q22-Bus only by posting a DMA request.

The MicroVAX accesses Q22-Bus memory and registers by using the following Q22-Bus cycles:

1. Data-Out-Byte (DATOB) for 8-bit writes
2. Data-In (DATI) for 16-bit reads (non-locked)
3. Data-Out (DATO) for 16-bit writes
4. Block Data In (DATBI) for 32-bit reads (non-locked)
5. Block Data Out (DATBO) for 32-bit writes
6. A DATI followed by a DATO for 16-bit read lock followed by a 16-bit write unlock
7. A DATBI followed by a DATBO for 32-bit read lock followed by a 32-bit write unlock

Note: When performing 32-bit reads from non-block mode memory, two successive DATI cycles are substitued for the DATBI cycle. When performing 32-bit writes to non-block mode memory, the two successive DATO cycles are substitued for the DATBO cycle. The same substitutions are made for the 32-bit read lock followed by a 32-bit write unlock. In all three cases, the KA630 retains control of the bus between successive DATI and/or DATO cycles.

Note: When the processor reads a byte from the Q-Bus the KA630-A performs a DATI cycle with address bit 0 correctly reflecting the byte address.

10.0 KA630 Boot and Diagnostic Facility

The KA630 Boot and Diagnostic Facility features one 16-bit register and two 28-pin ROM Sockets for 16K, 32K or 64K bytes of 16-bit read-only memory. The ROM memory is located on consecutive word boundaries and may be accessed via longword, word or byte references.

The KA630-A CPU Module populates the two ROM sockets with 64K bytes of 16-bit ROM (or EPROM). This ROM contains the KA630-A Console Program. If this ROM is replaced for special applications, the new ROM must contain some version of the Console Program.

10.1 Boot and Diagnostic Register (Hex Address: 2008 0000)

The 16-bit Boot and Diagnostic Register (BDR) is located at physical address 2008 0000. It can be accessed by KA630 software, but not by external Q22-Bus devices. The BDR allows the Boot and Diagnostic ROM programs to read various KA630 configuration bits and to load the 4-bit error display.

```
             1  1  1  1  1  1
             5  4  3  2  1  0  9  8  7  6  5  4  3  2  1  0
            ________________________________________________
           !  !  !     !  !  !  !  !              !  !  !  !  !
           !  !  ! 0 0 !  !  !  !  !  0 0 0 0  !  !  !  !  !
            -----------------------------------------------
             !  !        !  !  !  !              !  !  !  !
PWR OK   ----+  !        !  !  !  !              !  !  !  !
HLT ENB  -------+        !  !  !  !              !  !  !  !
                         !  !  !  !              !  !  !  !
CPU CD1  ----------------+  !  !  !              !  !  !  !
CPU CD0  -------------------+  !  !              !  !  !  !
                               !  !              !  !  !  !
BDG CD1  ----------------------+  !              !  !  !  !
BDG CD0  -------------------------+              !  !  !  !
                                                 !  !  !  !
DSPL 03  ----------------------------------------+  !  !  !
DSPL 02  -------------------------------------------+  !  !
DSPL 01  ----------------------------------------------+  !
DSPL 00  -------------------------------------------------+
```

Bit(s)	Mnemonic	Meaning
15	PWR OK	Power OK. This read-only bit is set if the Q22-Bus BPOK signal is asserted and clear if BPOK is negated.
14	HLT ENB	Halt Enable. This read-only bit reflects the status of external connector pin 13 (section 3.3.2). The set condition of this signal enables the various external halts. Also, following the execution of a HALT instruction in kernel mode, the KA630 ROM Program reads the HLT ENB bit to decide whether to enter the Console program (HLT ENB set) or to restart the operating system (HLT ENB clear).
13:12	-	Unused. Always Reads as zero.
11 10	CPU CD1 CPU CD0	CPU Code <01:00>. These two read-only bits originate from connector pins 14:15. They indicate whether the KA630 is configured as the arbiter or as one of the three auxiliaries: CPU CD <1:0> — Configuration 00 — KA630 Arbiter 01 — KA630 Auxiliary #1 10 — KA630 Auxiliary #2 11 — KA630 Auxiliary #3
09 08	BDG CD1 BDG CD0	Boot and Diagnostic Code <1:0>. This 2-bit read-only code reflects the status of configuration and display connector pins 14:13 (refer to section 3.3.2). The KA630 ROM programs use BDG CD <1:0> to determine the power up mode as follows: CPU CD <1:0> — Power Up Mode 00 — Run. 01 — Language Inquiry. 10 — Test. 11 — Manufacturing. Refer to section 10.1.1 for more details
07:04	-	Unused. Always Reads as zero.

Bit(s)	Mnemonic	Meaning
03	DSPL 03	Display <03:00>. These four write-only bits update an external LED display. Writing a "1" to a bit lights the corresponding LED. Writing a "0" to a bit turns its LED off. The display bits are set (all LED's are lit) by power up and by the negation of DCOK.
02	DSPL 02	
01	DSPL 01	
00	DSPL 00	

10.2 ROM Memory

10.2.1 ROM Sockets

The two ROM sockets are compatible with 8K, 16K and 32K X 8 ROM's. A machine-inserted jumper selects the pin 27 input to be either +5V (for 8K and 16K parts) or ROM Address bit 14 (for 32K parts).

The KA630-A is shipped with two 32K X 8 ROM's and with the jumper in the ROM Address bit 14 position.

10.2.2 ROM Address Space

The entire Boot and Diagnostic ROM may be read via either the 64KB Halt Mode ROM space or the 64KB Run Mode ROM space. Writes to either of these address spaces will result in a non-existant memory trap.

Any I-Stream Read from the Halt Mode ROM space places the KA630 in Halt Mode. The front panel RUN light is off and the Halt input to the MicroVAX chip is disabled.

Any I-Stream Read which does not access the Halt Mode ROM space, including reads from the Run Mode ROM space, places the KA630 in Run Mode. The front panel RUN light is lit and the Halt input to the MicroVAX chip is reenabled.

Writes and D-Stream Reads to any address space have no effect on Run Mode/Halt Mode status.

Note: I-Stream Reads include all instruction fetches (except when the MicroVAX chip retries a fetch following a non-existent memory or a parity error) and certain character string data fetches (again, except for those retries which follow an error). All reads which are not I-Stream Reads are D-Stream Reads.
A clear implication of this note is that, when running in Halt mode, the ROM programs can not use character string instructions to fetch data from outside the Halt Mode ROM address space.

10.2.2.1 Halt Mode ROM Address Space

The KA630 always responds to the full 64KB Halt Mode ROM Space (Hex Addresses: 2004 0000 - 2004 FFFF). When the KA630 contains 16KB of ROM memory, it appears four times, once within each 16KB of the Halt Mode ROM space. When the KA630 contains 32KB of ROM memory, it appears twice, once within each 32KB of ROM space.

10.2.2.2 Run Mode ROM Address Space

The KA630 always responds to the full 64KB Run Mode ROM space (Hex Addresses: 2005 0000 - 2005 FFFF). When the KA630 contains 16KB of ROM memory, it appears four times within the Run Mode ROM space. When the KA630 contains 32KB of ROM memory, it appears two times within the ROM memory space. Note that the Run Mode ROM space accesses the same ROM code as the Halt Mode ROM space.

10.2.3 KA630-A Console Program Operation

The KA630-A CPU Module populates the two ROM sockets with 64K bytes of 16-bit ROM (or EPROM). This ROM contains the KA630-A Console Program which can be entered by transferring program control to location 2004 0000.

Section 6.4.4 lists the various halt conditions which cause the MicroVAX to transfer program control to location 2004 0000. These conditions include the kernel mode halt instruction, assertion of the external halt input to the chip and certain fatal machine checks. When DCOK has been negated, either at power up time or by reboot, the combined assertion of DCOK and POK initiates program execution at location 2004 0000.

When running the KA630-A Console Program provides the services expected of a VAX-11 console system. In particular, the following services are available:

1. Automatic restart or bootstrap following processor halts or initial power up.

2. An interactive command language allowing the user to examine and alter the state of the processor.

3. Diagnostic tests executed on power up that check out the CPU, the memory system and the Q22-Bus Map.

4. Support of video or hardcopy terminals as the console terminal as well as support of QVSS based bitmapped terminals.

Refer to the KA630-A Console Program Specification for a complete description of these features.

10.2.3.1 Power Up Modes

The Boot and Diagnostic ROM programs use Boot and Diagnostic Code <1:0> (section 10.1) to determine the power up modes as follows:

Code	Mode
00	Run (factory setting). If the console terminal supports the Multi-national Character Set (MCS), the user will be prompted for language only if the time-of-year clock battery backup has failed. Full startup diagnostics are run.
01	Language Inquiry. If the console terminal supports MCS, the user will be prompted for language on every power up and restart. Full startup diagnostics are run.
02	Test. ROM programs run wrap-around serial line unit (SLU) tests.
03	Manufacturing. To provide for rapid startup during certain manufacturing test procedures, the ROM programs omit the power up memory diagnostics and set up the memory bit map on the assumption that all available memory is functional.

11.0 KA630 Time of Year Clock

11.1 Battery Backed-up Watch Chip

The KA630 contains the Motorola MC146818 CMOS watch chip and battery backup circuitry which interfaces, via the external connector, to a set of batteries which are mounted on the FCC cutout. The battery backup for this chip is spec'd to be greater than 240 hours when using three Ni Cad batteries in series.

The operating system software must fetch the correct time from this chip whenever power is restored to the system. If the power was off long enough for the battery voltage to go below spec, or if the battery was temporarily disconnected while the system power was off, the time in the watch chip is undefined. If the operating system detects a cleared Valid RAM and Time (VRT) bit (in watch chip register CSR D), it must prompt the system operator (or whoever turns on the system) for the time and then load this time into the watch chip.

Although the MicroVAX Interval Timer interrupts have a resolution of 10 ms, the watch chip only has a resolution of seconds. Therefore, the time resolution is as follows:

While under full power supply power .... 10 milliseconds

After power down for less than 240 hours
(while watch chip is powered by battery) .... 1 second

11.2 Watch Chip Registers

The watch chip contains 64 eight bit registers, 10 of which contain time of day data, 4 of which are CSRs, and the remaining 50 provide 50 bytes of battery backed-up RAM. They are addressed as follows from a base address of 200B 8000:

Number	Function	Address Offset from base address	Comments
0	seconds	00	used on reads only
1	second alarm	02	not used
2	minutes	04	loaded and read
3	minute alarm	06	not used
4	hours	08	loaded and read
5	hour alarm	0A	not used
6	day of week	0C	not used
7	date of month	0E	loaded and read
8	month	10	loaded and read
9	year	12	loaded to produce 28 or 29 day Feb. (not read by VMS)

Number	Function	Address Offset from base address	Comments
10	CSR A	14	loaded and read
11	CSR B	16	loaded and read
12	CSR C	18	not used
13	CSR D	1A	read-only
14	1st byte of RAM	1C	Uses assigned by the ROM code
	.		
	.		
63	50th byte of RAM	7E	

Note: Even though the addressing is on word boundaries, the time of year data and the RAM locations are loaded into or read out of the chip a byte at a time.

11.2.1 Time of Year Data Registers

Software should read the time of year data registers only after reading a cleared Update in Progress bit (CSR A <7>) and only when all interrupts are disabled (this assures that reading of the registers will not be delayed beyond the time for which they are valid). When the Update in Progress bit is clear, the content of these registers is guaranteed to be stable for 244 microseconds minimum.

Software should load the time of year data registers only after setting the SET bit (CSR B <7>). After loading the correct time and date into the time of year data registers, software loads 20 (hex) into CSR A and then clears the SET bit by loading 6 into CSR B.

ADDRESS	UNITS	DECIMAL RANGE	HEXADECIMAL RANGE
200B 8000	seconds	0 - 59	00 - 3B
200B 8004	minutes	0 - 59	00 - 3B
200B 8008	hours	0 - 23	00 - 17
200B 800E	day of mo.	1 - 31	01 - 1F
200B 8010	month	1 - 12	01 - 0C
200B 8012	year	0 - 99	00 - 63

11.2.2 Control and Status Register A

Control and Status Register A (CSR A) contains the Update in Progress bit (UIP), the divider selection bits (DV <2:0>) and the rate selection bits (RS <3:0>).

7	6	5	4	3	2	1	0
UIP	DV2 (0)	DV1 (1)	DV0 (0)	RS3 (0)	RS2 (0)	RS1 (0)	RS0 (0)

The UIP bit is a read only bit that is set when there is an update in progress within the chip. This bit must be read prior to reading the time. If the UIP bit is a 1, an update is in progress and the time registers are undefined. If the UIP bit is a 0, there is a minimum of 244 usec available prior to the next update cycle. (To completely read the 5 time registers should only require about 40 usec, worst case, if interrupts are disabled).

This register is undefined after battery power has been lost. Whenever the operating system software loads the time of year data registers, it must also load 20 (hex) into CSR A. Setting DV <2:0> = 2 sets up the timer for operation with the 32.768 khz oscillator. Setting RS <3:0> = 0 disables the unused interrupt and square wave outputs from the chip.

This register is not affected by the chip going into or out of the normal BBU mode, so long as the battery voltage remains within spec.

11.2.3 Control and Status Register B

Control and Status Register B (CSR B) contains the SET bit, four bits which enable chip functions which are not used in the KA630 design and three bits which control timer format and operation.

```
    7     6     5     4     3     2     1     0
+-----+-----+-----+-----+-----+-----+-----+-----+
! SET ! PIE ! AIE ! UIE ! SQWE! DM  !24/12! DSE !
!     ! (0) ! (0) ! (0) ! (0) ! (1) ! (1) ! (0) !
+-----+-----+-----+-----+-----+-----+-----+-----+
```

SET is a read-write bit that is used to enable and disable clock operation. When written with a 0, the internal time updates occur every second. When written with a 1, the updates are disabled so that the program may load the time of year registers. This bit must be set prior to setting the time. If the chip is in the middle of an update, setting this bit aborts the update.

Periodic Interrupt Enable (PIE), Alarm Interrupt Enable (AIE), Update Ended Interrupt Enable (UIE) and Square-Wave Enable (SQWE) are read-write bits which enable and disable chip outputs that are not used by the KA630 design.

Data Mode (DM) is a read-write bit which controls whether the time and date registers use binary or BCD formats. This bit is loaded with "1" to select binary format.

24/12 is a read-write bit which controls whether the hour register operates in 24-hour or 12-hour mode. This bit is loaded with a "1" to select 24-hour mode.

Daylight Savings Enable is a read-write bit which enables or disables special daylight savings time changes for the last Sunday in April and the last Sunday in October. This bit is loaded with a "0" to disable this function.

This register is undefined after battery power has been lost. Whenever the operating system software loads the time of year data registers, it must restart the timer by loading 6 (hex) into CSR B. Loading 6 into this register clears the SET bit and correctly loads the DM, 24/12 and Daylight Savings Time bits.

This register is not affected by the chip going into or out of the normal BBU mode, so long as the battery voltage remains within spec.

11.2.4 Control and Status Register C

This read-only register accesses interrupt flags which pertain to functions not implemented by the KA630 design.

11.2.5 Control and Status Register D

Control and Status Register D (CSR D) is read-only register which contains the Valid RAM and Time bit (VRT). The remaining seven bits always read as zeros.

```
     7     6     5     4     3     2     1     0
  +-----+-----+-----+-----+-----+-----+-----+-----+
  | VRT |              Read as zeros              |
  +-----+-----+-----+-----+-----+-----+-----+-----+
```

The Valid RAM and Time (VRT) bit is read by the software, before reading the time registers, to verify the validity of the time. If the battery voltage goes below spec during the BBU mode, this bit is set to 0 by the hardware sensing circuitry during powerup, indicating that the time registers are undefined. The VRT bit is automatically set when this register is read. If this bit was clear, the time registers must be updated immediately, since this bit will subsequently indicate that the chip contains a valid time setting.

Note: If battery voltages are removed and then restored during power down, KA630 logic guarantees that VRT = 0.

11.2.6 RAM Memory

The fifty bytes of RAM memory are used by the KA630-A Console Program to store information required to restart the machine following a Halt which transfers program control to location 20040000 (hex). One of these RAM locations, designated the Console Program Mailbox (CPMBX), is used for communication between the operating system and the console program.

The use of these bytes is defined in the KA630-A Console Program Specification.

11.3 Powerup

Following a power up, the KA630 console program reads the VRT bit in CSR D. If this bit is set, the RAM and time data are valid. If this bit is clear, then the RAM and time data are invalid, and the console program disables the clock by setting the CSR B SET bit.

When the operating system gains control of the machine, it checks the CSR B SET bit. If that bit is set, then the operating system must request the correct time of year from the operator (or from whoever turns on the system).

11.3.1 Valid RAM and Time

If the VRT bit is set, then the RAM and Time data is valid. The operating system reads the UIP bit in CSR A, to assure that an update isn't in progress. If this bit is read as a 1, the watch chip is doing an update and the data is invalid until it is through. The maximum time for the update is 1.984 ms.

If the UIP bit is read as a 0, then the clock registers can be read by the operating system. The operating system reformats the time into a 32-bit count and loads it into the memory location which contains the time of day count during system operation.

11.3.2 Invalid RAM and Time

If the VRT bit is clear, then the RAM and Time data is invalid. The operating system stops timer operation by setting the SET bit (CSR B <7>) and then requests the time of year from the operator. After loading the correct time and date into the time of year data registers, the operating system loads 10 (hex) into CSR A and then clears the SET bit by loading 6 into CSR B.

The operating system also reformats the time into a 32-bit count and loads it into the memory location which contains the time of day count during system operation.

12.0   Interval Timer

The KA630 Interval Timer is contained within the MicroVAX chip. When it is enabled, the interval timer posts an interrupt request every 10 msec.

12.1   Interval Clock Control/Status Register   (IPR 24)

The Interval Clock Control/Status Register (ICCS) is accessed as internal processor register 24 (decimal). ICCS implementation is unique to the MicroVAX Chip and consists of a minimal interval timer control.

```
 3
 1                                             7 6 5            0
+-----------------------------------------------+-+-------------+
!                unused, returns 0              ! ! 0 0 0 0 0 0 !
+-----------------------------------------------+-+-------------+
                                                 !
                                                 !
Interrupt Enable (IE)   -------------------------+
```

ICCS <6> (IE) is a read-write bit which enables and disables the interval timer interrupts. When this bit is set, an interval timer interrupt is requested every 10 msec. When ICCS <6> is clear, interval timer interrupts are disabled. ICCS <6> is cleared by power up and by the negation of DCOK.

12.2   Interval Timer Operation

When ICCS <6> is set, the interval timer posts an interrupt request every 10 msec. The interval timer is the highest priority device at interrupt priority level 16 (hex). The interrupt vector for the interval timer is C0 (hex).

13.0 KDQ11 Console Serial Line Unit

13.1 Console Functionality

The console serial line provides the KA630 processor with a serial interface for the console terminal. The console serial line is full duplex. It provides an RS-423 EIA interface which is also RS-232C compatible.

This serial line interface is based on the DC319 Digital Link Asynchronous Receiver Transmitter (DLART), described in Digital Purchase Specification A-PS-2117312-0-0.

The receive and transmit baud rates are always identical and are determined by the Baud Rate Select signals (BRS <02:00> L) which are received from an external 8-position switch via a connector mounted at the top of the module.

The Baud Rate is selected as follows:

BRS02 L	BRS01 L	BRS00 L	Baud Rate
H	H	H	300
H	H	L	600
H	L	H	1200
H	L	L	2400
L	H	H	4800
L	H	L	9600
L	L	H	19200
L	L	L	38400

13.2 Console Registers

There are four registers associated with the Console Serial Line Unit. They are accessed via internal processor registers 32-35 (decimal), per section 6.2.3:

Number	Register Name	Mnemonic
32	Console Receiver Control/Status	RXCS
33	Console Receiver Data Buffer	RXDB
34	Console Transmit Control/Status	TXCS
35	Console Transmit Data Buffer	TXDB

13.2.1 Console Receiver Control/Status Register (IPR 32)

```
 3                                    1 1 1
 1                                    2 1 0   8 7 6 5            0
+------------------------------------+-+-----+-+-+--------------+
!          unused, returns 0         ! !0 0 0! ! ! 0 0 0 0 0 0  !
+------------------------------------+-+-----+-+-+--------------+
                                       !       ! !
                RCV ACT     -----------+       ! !
                                               ! !
                RX DONE     -------------------+ !
                RX IE       ---------------------+
```

Bit(s)	Mnemonic	Meaning
31:12	-	Unused. Read as zeros.
11	RCV ACT	Receiver Active. This read-only bit is set at the center of the start bit of the serial input data and is cleared at the expected center (per DLART timing) of the stop bit at the end of the serial data. RX DONE is set one bit time after RCV ACT clears.
10:08		Unused. Read as zeros.
07	RX DONE	Receiver Done. This read-only bit is set when an entire character has been received and is ready to be read from the RBUF Register. This bit is automatically cleared when RBUF is read. It is also cleared by power up, by the negation of DCOK and by writes to the Bus Initialize Register.
06	RX IE	Receiver Interrupt Enable. This read-write bit is cleared by power up, by the negation of DCOK and by writes to the Bus Initialize Register. If RX DONE and RX IE are both set, a program interrupt is requested.
05:00		Unused. Read as zeros.

13.2.2 Console Receiver Data Buffer (IPR 33)

```
 3                           1 1 1 1 1 1 1
 1                           6 5 4 3 2 1 0   8 7                0
+----------------------------+-+-+-+-+-+-+-----+-----------------+
!      unused, returns 0     ! ! ! !0! !0 0 0!                  !
+----------------------------+-+-+-+-+-+-+-----+-----------------+
                              ! ! !   !               !
                              ! ! !   !               !
                ERR  ---------+ ! !   !               !
                OVR ERR  -------+ !   !               !
                FRM ERR  ---------+   !               !
                                      !               !
                RCV BRK  -------------+               !
                                                      !
                Received Data Bits  ------------------+
```

Bit(s)	Mnemonic	Meaning
31:16	-	Unused. Always read as zero.
15	ERR	Error. This read-only bit is set if RBUF <14> or <13> is set. ERR is clear if these two bits are clear. This bit cannot generate a program interrupt.
14	OVR ERR	Overrun Error. This read-only bit is set if a previously received character was not read before being overwritten by the present character.
13	FRM ERR	Framing Error. This read-only bit is set if the present character had no valid stop bit.

NOTE: Error conditions remain present until the next character is received, at which point, the error bits are updated. The Error bits are cleared by power up and by the negation of DCOK.

Bit(s)	Mnemonic	Meaning
12		Unused. This bit always reads as "0".
11	RCV BRK	Received Break. This read-only bit is set at the end of a received character for which the serial data input remained in the SPACE condition for all 11 bit times. RCV BRK then remains set until the serial data input returns to the MARK condition. RCV BRK is also cleared by power up and by the negation of DCOK.
10:08		Unused. These bits always read as "0".

Bit(s)	Mnemonic	Meaning
07:00		Received Data Bits. These read-only bits contain the last received character.

13.2.3 Console Transmitter Control/Status Register (IPR 34)

```
 3
 1                                           8 7 6 5   3 2 1 0
+-------------------------------------------+-+-+-----+-+-+-+
!            unused, returns 0              ! ! !0 0 0! !0! !
+-------------------------------------------+-+-+-----+-+-+-+
                                             ! !       !   !
               TX RDY  ----------------------+ !       !   !
                                               !       !   !
               TX IE   ------------------------+       !   !
                                                       !   !
               MAINT   --------------------------------+   !
                                                           !
               XMIT BRK  ----------------------------------+
```

Bit(s)	Mnemonic	Meaning
31:08		Unused. Read as zeros.
07	TX RDY	Transmitter Ready. This read-only bit is cleared when XBUF is loaded and sets when XBUF can receive another character. XMT RDY is set by power up, by the negation of DCOK and by writes to the Bus Initialize Register.
06	TX IE	Transmitter Interrupt Enable. This read-write bit is cleared by power up, by the negation of DCOK and by writes to the Bus Initialize Register. If both TX RDY and TX IE are set, a program interrupt is requested.
05:03		Unused. Read as zeros.
02	MAINT	Maintenance. This read-write bit is used to facilitate a maintenance self-test. When MAINT is set, the external serial input is disconnected and the serial output is used as the serial input. This bit is cleared by power up, by the negation of DCOK and by writes to the Bus Initialize Register.
01		Unused. Read as zero.

Bit(s)	Mnemonic	Meaning
00	XMIT BRK.	Transmit Break. When this read-write bit is set, the serial output is forced to the SPACE condition. XMIT BRK is cleared by power up, by the negation of DCOK and by writes to the Bus Initialize Register.

13.2.4 Console Transmitter Data Buffer (IPR 35)

XBUF bits <31:08> are not used. XBUF bits <7:0> are write-only bits used to load the transmitted character.

13.3 Additional Specifications

Serial Data Format:	8-bit data, No Parity, One Stop Bit
Interrupt Vectors:	F8 (hex) Receiver FC (hex) Transmitter
Interrupt Priority:	IPL 14 (hex); same as BR4 on Q-Bus (the console serial line unit is the highest priority BR4 device)

Note: Following a console terminal interrupt, the KA630 CPU sets the IPL = 14. The IPL is set = 17 for external Q22-Bus BR4 interrupts.

13.4 Break Response

The KA630 Console Serial Line Unit may be configured either to perform a halt operation or to have no response when a break condition is received. A halt operation will cause the processor to transfer program control to ROM location 2004 0000 (hex).

The Halt on Break Option is enabled if the Connector HLT ENB signal is asserted (refer to sections 3.3.2 and 10.1).

The DLART recognizes a break condition at the end of a received character for which the serial data input remained in the SPACE condition for all 11 bit times. The Break Recognition line remains asserted until software reads the RBUF.

14.0 Q22-Bus Control

14.1 Bus Initialize Register (IPR 55)

The Bus Initialize Register is accessed as internal processor register 55 (decimal). On an arbiter KA630, writing to this register asserts the Q22-Bus BINIT signal for 10 usec (+/- 20%) and clears all on-board register bits which are specified as being cleared by writes to the Bus Initialize Register. On an auxiliary KA630, writing to this register does not assert the Q22-Bus BINIT signal, but it does clear all on-board register bits specified as being cleared by writes to the Bus Initialize Register. For either configuration (arbiter or auxiliary), this register always reads as zero.

Note: An auxiliary KA630 module receives BINIT from the Q22-Bus and uses that signal to initialize the MicroVAX chip and to clear all internal register bits which are specified as being cleared by the negation of DCOK (i.e. the assertion of the Q22-Bus BINIT signal has the same effect on auxiliary modules as the negation of DCOK).

14.2 Multi-level Interrupts

When the KA630 is configured as the arbiter CPU, it responds to interrupt requests BR7-4 with the standard Q22-Bus interrupt acknowledge prototcol (DIN followed by IAK). The console serial line unit and the interprocessor doorbell can request interrupts at BR level 4 and have priority over all Q22-Bus BR4 interrupt requests. After responding to any interrupt request BR7-4, the MicroVAX CPU sets the processor priority to IPL 17. All BR7-4 interrupt requests are disabled unless software lowers the processor priority.

When the KA630 is configured as an auxiliary, it does not respond to interrupt requests from the Q22-Bus. However, it does respond to the BR4 level interrupt requests from its console serial line unit and interprocessor doorbell.

Interrupt requests from the KA630 interval timer are handled internally by the MicroVAX chip. Interval timer interrupt requests have a higher priority than BR6 interrupt requests. After responding to an interval timer interrupt request, the MicroVAX CPU sets the processor priority to IPL 16. Thus, BR7 interrupt requests remain enabled.

14.3 Interprocessor Communications Facility

The KA630 Interprocessor Communication Facility allows other processors on the system to request program interrupts from the KA630 without using the Q22-Bus interrupt request lines. It also controls external access to local memory (via the Q22-Bus map) and allows other processors to "halt" an auxiliary CPU.

14.3.1 Interprocessor Communication Register

The Interprocessor Communication Register resides in the Q22-Bus I/O page address space and can be accessed by any device which can become Q22-Bus master (including the KA630 itself). The ICR is byte accessible, meaning that a write byte instruction can write to either the low or high byte without affecting the other byte.

The I/O Page address of the ICR varies with the four configurations of arbiter and auxiliary KA630:

Hex 32-Bit Address	Octal 22-Bit Address	Register
2000 1F40	17 777 500	ICR (KA630 Arbiter CPU)
2000 1F42	17 777 502	ICR (KA630 Auxiliary #1)
2000 1F44	17 777 504	ICR (KA630 Auxiliary #2)
2000 1F46	17 777 506	ICR (KA630 Auxiliary #3)

```
          1   1   1   1   1   1
          5   4   3   2   1   0   9   8   7   6   5   4   3   2   1   0
        +---+-------------------+---+---+---+---+---------------+---+
        !   !   0 0 0 0 0 0     !   !  0!   !   !  0 0 0 0      !   !
        +---+-------------------+---+---+---+---+---------------+---+
          !                         !       !   !                   !
DMA QPE  ---+                        !       !   !                   !
                                    !       !   !                   !
AUX HLT  ---------------------------+       !   !                   !
                                            !   !                   !
DBI IE   -----------------------------------+   !                   !
LM EAE   ---------------------------------------+                   !
                                                                    !
DBI RQ   -----------------------------------------------------------+
```

Bit(s)	Mnemonic	Meaning
15	DMA QPE	DMA Q22-Bus Address Space Parity Error. This read-only bit is set if Memory System Error Register bit <04> (DMA QPE) is set. The DMA QPE bit indicates that a parity error occured when an external device (or CPU) was accessing the KA630 local memory.
14:09	-	Unused. Read as zeros.
08	AUX HLT	Auxiliary Halt. On an auxiliary KA630, AUX HLT is a read-write bit. When set, typically by the arbiter CPU, it causes the on-board CPU to transfer program control to the Halt Mode ROM Code. On an arbiter KA630, AUX HLT is a read-only bit which always reads as zero. It has no effect on arbiter CPU operation.

Bit(s)	Mnemonic	Meaning
07	-	Unused. Read as zero.
06	DBI IE	Doorbell Interrupt Enable. This bit, when set, enables interprocessor doorbell interrupt requests via ICR <00>. When the on-board CPU is Q22-Bus master, DBI IE is a read-write bit. When an external device (or CPU) is bus master, DBI IE is a read-only bit. DBI IE is cleared by power up, by the negation of DCOK and by writes to the Bus Initialize Register.
05	LM EAE	Local Memory External Access Enable. This bit, when set, enables external access to local memory (via the Q22-Bus map). When the on-board CPU is Q22-Bus master, LM EAE is a read-write bit. When an external device (or CPU) is bus master, LM EAE is a read-only bit. LM EAE is is cleared by by power up, by the negation of DCOK and by writes to the Bus Initialize Register.
04:01	-	Unused. Read as zeros.
00	DBI RQ	Doorbell Interrupt Request. If ICR <06> (DBI IE) is set, writing a "1" to DBI RQ sets DBI RQ, thus requesting a doorbell interrupt. If ICR <06> is clear, writing a "1" to DBI RQ has no effect. Writing a "0" to DBI RQ has no effect. DBI RQ is cleared when the CPU grants the doorbell interrupt request. DBI RQ is held clear whenever DBI IE is clear.

14.3.2 Interprocessor Doorbell Interrupts

If the Interprocessor Communication Register DBI IE bit is set, any Q22-Bus Master can request an interprocessor doorbell interrupt by writing a "1" into ICR bit <00>.

Interrupt Vector:	204 (hex)
Interrupt Priority:	IPL 14 (hex); same as BR4 on Q-Bus (the interprocessor doorbell is the second highest priority BR4 device, directly after the console serial line unit).

Note: Following a interprocessor doorbell interrupt, the KA630 CPU sets the IPL = 14. The IPL is set = 17 for external Q22-Bus BR4 interrupts.

15.0 Multi-Processor Considerations

15.1 Auxiliary/Arbiter Differences

When the KA630 is configured as an auxiliary, its operation differs from operation as an arbiter KA630 in several important areas:

1. The arbiter KA630 arbitrates bus mastership per the Q22-Bus DMA protocol; the arbitration logic is disabled on an auxiliary KA630.

2. Both the arbiter and auxiliary KA630 request bus mastership via the Q22-Bus DMA Request protocol.
    a. They both assert BDMR on the Q22-Bus.
    b. The arbiter KA630 receives DMGI from its arbitration logic; the auxiliary receives DMGI from its Q22-Bus BDMGI pin.
    c. Only the auxiliary KA630 actually asserts BSACK on the Q22-Bus.

3. The arbiter KA630 asserts the Q22-Bus BINIT signal when DCOK is negated and when its CPU software writes to its Bus Initialize Register; the auxiliary KA630 never asserts Q22-Bus BINIT, but receives BINIT and uses it to initialize the MicroVAX chip and to clear all internal registers which are cleared by the negation of DCOK. (Refer to section 14.1).

4. The physical address of the Interprocessor Communication Register is different for each of the four KA630 arbiter/auxiliary configurations (per section 14.3.1).

5. An auxiliary KA630 can be halted by setting bit <08> (AUX HLT) of its Interprocessor Communication Register. On an arbiter KA630, this feature is disabled and AUX HLT is a read-only bit which always reads as zero. (Refer to section 14.3.1).

6. The CPU halts are controlled by the external connector HLT ENB input. However, the external halts which are affected differ somewhat for the arbiter and auxiliary KA630 modules. (Refer to section 3.3.2).

7. Each arbiter or auxiliary KA630 module can field interrupt requests from its interval timer, from its console device, and from its interprocessor doorbell. Only the arbiter KA630 can field interrupts from Q22-Bus interrupt request lines BR7-4.

8. The arbiter asserts BIAKO to the Q22-Bus when it responds to a Q22-Bus interrupt request; the auxiliary asserts BIAKO to the Q22-Bus when it receives the assertion of BIAKI from the Q22-Bus.

9. Although both the arbiter and auxiliary KA630 modules contain the same time of year clock and battery back-up circuitry, it is assumed that the auxiliary will be configured without batteries and that its clock will never actually be enabled.

15.2 Multi-Processor Features

The following features have been added to the KA630 to allow its use in multi-processor systems:

1. A 2-bit code, received at the external connector, allows the KA630 module to be configured as the arbiter or as one of three auxiliaries (section 3.3.2). Section 15.1 presents a list of arbiter/auxiliary differences.

2. The Interprocessor Communication Register (section 14.3) provides a mechanism for interprocessor interrupts, for enabling and disabling external access to local memory and for flagging local memory parity errors caused by external references. On auxiliary KA630 modules, it also provides a mechanism for "halting" the CPU.

15.3 KA630 Based Multi-Processor Systems

The KA630 multi-processor features were designed for use in a message passing environment similar to the System Communications Architecture (SCA) which is currently layered on the CI Port Architecture.

Each KA630 processor in a system fetches instructions and data primarily from its own local memory. The various processors communicate via message queues stored in local memory which has been mapped to the Q22-Bus address space. Typically, the processors use the interprocessor doorbell feature to interrupt each other after placing a message in an empty queue.

In most systems all Q22-Bus devices would be under the direct control of the arbiter processor which fields all interrupts. When a disk controller is under the direct control of the arbiter CPU, then the arbiter must set up the transfer of program and data information between the corresponding disks and the auxiliary processors. The auxiliary processor would be responsible for setting up its own Q22-Bus Map to point to the local memory space which is a target of that transfer.

Following a power up or system restart, the auxiliary CPU runs its self-test diagnostics, clears the valid bits in its Q22-Bus Map mapping registers, enters Halt Mode ROM space and then sets its own Interprocessor Communication Register bits <08> (AUX HLT) and <06:05> (DBI IE and LM EAE). The arbiter CPU waits for the auxiliary's LM EAE bit to set, "boots" the auxiliary CPU by loading the appropriate programs and data into the arbiter's own local memory which are mapped (via the Q22-Bus map) to an assigned Q22-Bus address space. The arbiter then clears the auxiliary's AUX HLT bit. The auxiliary CPU, still in Halt Mode ROM space, waits for its AUX HLT bit to clear and then begins auxiliary execution at a specified location in the Q22-Bus address space (referencing local memory in the arbiter).

15.4 PDP-11 Based Multi-Processor Systems

Up to three auxiliary KA630 modules may be added to a KDF11-B or KDJ11-B based Q22-Bus system. Operation of a PDP-11 based system is similar to that of a KA630 based system. However, the following issues must be addressed:

1. When a PDP-11 processor is arbiter, its "local" memory is actually Q22-Bus memory. This appears to present no special problems. Obviously, a portion of the Q22-Bus memory address space must be reserved for mapping the auxiliary KA630 modules' local memory.

2. Since the PDP-11 does not contain an interprocessor communication register, an external device must be added which allows the auxiliary KA630 modules to interrupt the PDP-11.

   /Since the KA630 console program does not interrupt the arbiter CPU, they do not require modification if this external device is not compatible with the KA630 interprocessor communication register (one could use either a DLV11 or a DRV11)./

APPENDIX A
Module Pinouts

A.1 KA630 Module Pinouts

The KA630 AB row module pinouts are compatible with the Q22-Bus specification (DEC standard 160). The SRUN L signal appears on pin AF1. The CD row module pinouts utilize the CD-Interconnect to communicate with up to two memory expansion modules.

Note: The KA630 Module can not be used in slots for which the Q22-Bus is connected to both the AB and CD rows (Q22/Q22 configuration). The backplane CD rows must be compatible with the entire CD-Interconnect specification (required for the use of MS630 modules) or must make no connections except for the +5 volt and ground pins designated by the CD-Interconnect specification.

Tables A-1 and A-2 list the KA630 AB and CD slot pinouts.

AA1	BIRQ5 L	AA2	+5	BA1	BDCOK H	BA2	+5
AB1	BIRQ6 L	AB2		BB1	BPOK H	BB2	
AC1	BDAL16 L	AC2	GND	BC1	BDAL18 L	BC2	GND
AD1	BDAL17 L	AD2	+12	BD1	BDAL19 L	BD2	+12
AE1		AE2	BDOUT L	BE1	BDAL20 L	BE2	BDAL02 L
AF1	SRUN L	AF2	BRPLY L	BF1	BDAL21 L	BF2	BDAL03 L
AH1		AH2	BDIN L	BH1		BH2	BDAL04 L
AJ1	GND	AJ2	BSYNC L	BJ1	GND	BJ2	BDAL05 L
AK1		AK2	BWTBT L	BK1		BK2	BDAL06 L
AL1		AL2	BIRQ4 L	BL1		BL2	BDAL07 L
AM1	GND	AM2	BIAKI L	BM1	GND	BM2	BDAL08 L
AN1	BDMR L	AN2	BIAKO L	BN1	BSACK L	BN2	BDAL09 L
AP1	BHALT L	AP2	BBS7 L	BP1	BIRQ7 L	BP2	BDAL10 L
AR1	BREF L	AR2	BDMGI L	BR1	BEVNT L	BR2	BDAL11 L
AS1		AS2	BDMGO L	BS1		BS2	BDAL12 L
AT1	GND	AT2	BINIT L	BT1	GND	BT2	BDAL13 L
AU1		AU2	BDAL00 L	BU1		BU2	BDAL14 L
AV1		AV2	BDAL01 L	BV1	+5	BV2	BDAL15 L

KA630 Module AB Slot Pinouts
Table A-1

CA1		CA2	+5	DA1		DA2	+5
CB1		CB2	MAA<9> L	DB1		DB2	MAA<7> L
CC1		CC2	GND	DC1		DC2	GND
CD1		CD2	RAS<5> H	DD1		DD2	MAA<5> L
CE1		CE2	BMCAS<0> L	DE1		DE2	MAA<4> L
CF1		CF2	RAS<1> H	DF1		DF2	MAA<3> L
CH1		CH2	BMCAS<1> H	DH1		DH2	MAA<6> L
CJ1		CJ2	MSID<0> L	DJ1		DJ2	MSID<2> L
CK1		CK2	MSWT<1> H	DK1		DK2	RAS<3> H
CL1		CL2	RAS<4> H	DL1		DL2	RAS<7> H
CM1		CM2	MSID<1> L	DM1		DM2	MSID<3> L
CN1		CN2	MAA<1> L	DN1		DN2	RAS<2> H
CP1		CP2	MAA<2> L	DP1		DP2	BMCAS<2> H
CR1		CR2	MAA<0> L	DR1		DR2	BMCAS<3> H
CS1		CS2	MAA<8> L	DS1		DS2	
CT1	GND	CT2	MSID<4> L	DT1	GND	DT2	
CU1		CU2	RAS<0> H	DU1		DU2	RAS<6> H
CV1		CV2		DV1		DV2	

KA630 Module CD Slot Pinouts
Table A-2

A.2 MS630 Module Pinouts

The MS630-AA is a dual height module which mounts in the CD rows and, therefore, has CD Row pinouts only. The MS630-BA, MS630-BB and MS630-CA are quad height modules which have both AB and CD row pinouts.

The MS630 AB row module pinouts connect with +5 volts (pins AA2, BA2 and BV1) and ground (pins AC2, AJ1, AM1, AT1, BC2, BJ1, BM1 and BT1) only. The MS630 also connects pin AM2 to pin AN2 (passing BIAK) and pin AR2 to AS2 (passing BDMG). The CD row module pinouts require the CD-Interconnect to communicate with the KA630 module and/or another MS630 module.

Note: The MS630 Module can not be used in slots for which the Q22-Bus is connected to both the AB and CD rows (Q22/Q22 configuration). The CD rows must be compatible with the the CD-Interconnect specification.

Tables A-3 and A-4 list the MS630 AB and CD slot pinouts.

AA1		AA2	+5	BA1		BA2	+5
AB1		AB2		BB1		BB2	
AC1		AC2	GND	BC1		BC2	GND
AD1		AD2		BD1		BD2	
AE1		AE2		BE1		BE2	
AF1		AF2		BF1		BF2	
AH1		AH2		BH1		BH2	
AJ1	GND	AJ2		BJ1	GND	BJ2	
AK1		AK2		BK1		BK2	
AL1		AL2		BL1		BL2	
AM1	GND	AM2	BIAK L	BM1	GND	BM2	
AN1		AN2	BIAK L	BN1		BN2	
AP1		AP2		BP1		BP2	
AR1		AR2	BDMG L	BR1		BR2	
AS1		AS2	BDMG L	BS1		BS2	
AT1	GND	AT2		BT1	GND	BT2	
AU1		AU2		BU1		BU2	
AV1		AV2		BV1	+5	BV2	

MS630 Module AB Slot Pinouts
Table A-3

CA1		CA2	+5	DA1		DA2	+5
CB1	MAA<9> L	CB2	MAA<9> L	DB1	MAA<7> L	DB2	MAA<7> L
CC1		CC2	GND	DC1		DC2	GND
CD1	RAS<5> H	CD2	RAS<1> H	DD1	MAA<5> L	DD2	MAA<5> L
CE1	BMCAS<0> H	CE2	BMCAS<0> L	DE1	MAA<4> L	DE2	MAA<4> L
CF1	RAS<1> H	CF2		DF1	MAA<3> L	DF2	MAA<3> L
CH1	BMCAS<1> H	CH2	BMCAS<1> H	DH1	MAA<6> L	DH2	MAA<6> L
CJ1	MSID<0> L	CJ2	MSID<2> L	DJ1	MSID<2> L	DJ2	
CK1	MSWT<1> H	CK2	MSWT<1> H	DK1	RAS<3> H	DK2	
CL1	RAS<4> H	CL2	RAS<0> H	DL1	RAS<7> H	DL2	RAS<3> H
CM1	MSID<1> L	CM2	MSID<3> L	DM1	MSID<3> L	DM2	
CN1	MAA<1> L	CN2	MAA<1> L	DN1	RAS<2> H	DN2	
CP1	MAA<2> L	CP2	MAA<2> L	DP1	BMCAS<2> H	DP2	BMCAS<2> H
CR1	MAA<0> L	CR2	MAA<0> L	DR1	BMCAS<3> H	DR2	BMCAS<3> H
CS1	MAA<8> L	CS2	MAA<8> L	DS1	Spare	DS2	Spare
CT1	GND	CT2	MSID<4> L	DT1	GND	DT2	MSID<4> L
CU1	RAS<0> H	CU2		DU1	RAS<6> H	DU2	RAS<2> H
CV1		CV2		DV1		DV2	

MS630 Module CD Slot Pinouts
Table A-4

APPENDIX B
Indicators, Switches, and Jumpers

B.1 KA630 Indicator Lights

Five Light Emitting Diodes are mounted at the top of KA630 Module. The single green LED is lit if the Q-Bus BDCOK signal is asserted. The four red LED's are controlled from the Boot and Diagnostic Display Register. The status of these four bits is also available on the connector mounted at the top of the KA630 module.

B.2 Switches

The KA630 contains no switches. All configuration information is received via the external connector described in section 3.3.2.

B.3 Manufacturing Test Jumpers

To be supplied.

APPENDIX C
Physical Address Assignments

C.1 General

Address Range			Contents
0000 0000	-	00FF FFFF	Local Memory Space (16MB)
0100 0000	-	1FFF FFFF	Reserved Memory Space (496MB)
2000 0000	-	2000 1FFF	Q22-Bus I/O Space (8KB)
2000 2000	-	2003 FFFF	Reserved I/O Space (248KB)
2004 0000	-	2004 FFFF	Halt Mode ROM Space (64KB)
2005 0000	-	2005 FFFF	Run Mode ROM Space (64KB)
2006 0000	-	2007 FFFF	Reserved Local ROM Space (128KB)
2008 0000	-	200B FFFF	Local Register I/O Space (256KB)
200C 0000	-	3FFF FFFF	Reserved I/O Space (255.25 MB)
3000 0000	-	303F FFFF	Q22-Bus Memory Space (4MB)
3040 0000	-	3FFF FFFF	Reserved I/O Space (252MB)

C.2 Q22-Bus I/O Space

The only KA630 register located in the 8KB Q22-Bus I/O address space (2000 0000 - 2000 1FFF) is the Interprocessor Communication Registers. This register is accessible by both the KA630 CPU and external Q22-Bus devices:

Hex 32-Bit Address	Octal 22-Bit Address	Register
2000 1F40	17 777 500	ICR (KA630 Arbiter CPU)
2000 1F42	17 777 502	ICR (KA630 Auxiliary #1)
2000 1F44	17 777 504	ICR (KA630 Auxiliary #2)
2000 1F46	17 777 506	ICR (KA630 Auxiliary #3)

C.3 Local Register I/O Space

The 256KB Local Register I/O Space (2008 0000 - 200B FFFF) contains KA630 registers which are not accessible from the Q22-Bus. Most of the addresses in this space are unassigned and respond as non-existent memory.

The following KA630 registers have device addresses within the Local Register I/O Space:

Hex 32-Bit Address	Register
2008 0000	Boot and Diagnostic Register
2008 0004	Memory System Error Register
2008 0008	CPU Error Address Register
2008 000C	DMA Error Address Register
2008 8000	Q22-Bus Map Registers
.	.
.	.
.	.
2008 FFFC	.
200B 8000	Time of Year Clock Registers
.	.
.	.
.	.
200B 807E	.