VAX 6000—400

digital

Mini-Reference

Order Number: EK-640EA-HR-001

VAX 6000–400 Mini-Reference

Order Number EK–640EA–HR–001

This manual supplies easy-to-access key information on VAX 6000–400 systems.

digital equipment corporation
maynard, massachusetts

First Printing, July 1989

Contents

Examples

Figures

Tables

Preface

Intended Audience

This manual is intended for the system manager and programmer.

Document Structure

This manual has six chapters:

- **Chapter 1—Console Operation**
- **Chapter 2—Self-Test**
- **Chapter 3—Address Space**
- **Chapter 4—KA64A CPU Module Registers**
- **Chapter 5—MS62A Memory Registers**
- **Chapter 6—DWMBA Adapter Registers**

VAX 6000–400 Documents

Documents in the VAX 6000–400 documentation set include:

Title	Order Number
VAX 6000–400 Installation Guide	EK–640EA–IN
VAX 6000–400 Owner's Manual	EK–640EA–OM
VAX 6000–400 Mini-Reference	EK–640EA–HR
VAX 6000–400 System Technical User's Guide	EK–640EA–TM
VAX 6000–400 Options and Maintenance	EK–640EA–MG
VAX 6000 Series Upgrade Manual	EK–600EA–UP

Associated Documents

Other documents that you may find useful include:

Title	Order Number
CIBCA User Guide	EK–CIBCA–UG
DEBNI Installation Guide	EK–DEBNI–IN
Guide to Maintaining a VMS System	AA–LA34A–TE
Guide to Setting Up a VMS System	AA–LA25A–TE
HSC Installation Manual	EK–HSCMN–IN
H4000 DIGITAL Ethernet Transceiver Installation Manual	EK–H4000–IN
H7231 Battery Backup Unit User's Guide	EK–H7231–UG
Installing and Using the VT320 Video Terminal	EK–VT320–UG
Introduction to VMS System Management	AA–LA24A–TE
KDB50 Disk Controller User's Guide	EK–KDB50–UG
RA90 Disk Drive User Guide	EK–ORA90–UG
RV20 Optical Disk Owner's Manual	EK–ORV20–OM
SC008 Star Coupler User's Guide	EK–SC008–UG
TK70 Streaming Tape Drive Owner's Manual	EK–OTK70–OM
TU81/TA81 and TU81 PLUS Subsystem User's Guide	EK–TUA81–UG
ULTRIX–32 Guide to System Exercisers	AA–KS95B–TE
VAX Architecture Reference Manual	EY–3459E–DP
VAX Systems Hardware Handbook — VAXBI Systems	EB–31692–46
VAXBI Expander Cabinet Installation Guide	EK–VBIEA–IN
VAXBI Options Handbook	EB–32255–46
VMS Installation and Operations: VAX 6000 Series	AA–LB36B–TE
VMS Networking Manual	AA–LA48A–TE
VMS System Manager's Manual	AA–LA00A–TE
VMS VAXcluster Manual	AA–LA27A–TE

Chapter 1
Console Operation

This chapter provides reference information for working at the console terminal.

Terminal setup characteristics:

- The maximum recommended baud rate is 1200.

  If the console is not responding, you may need to press the Break key to increment the baud rate.

- Terminal characteristics should be set to the following: eight bits, no parity, one stop bit.

Figure 1–1: International and English Control Panels

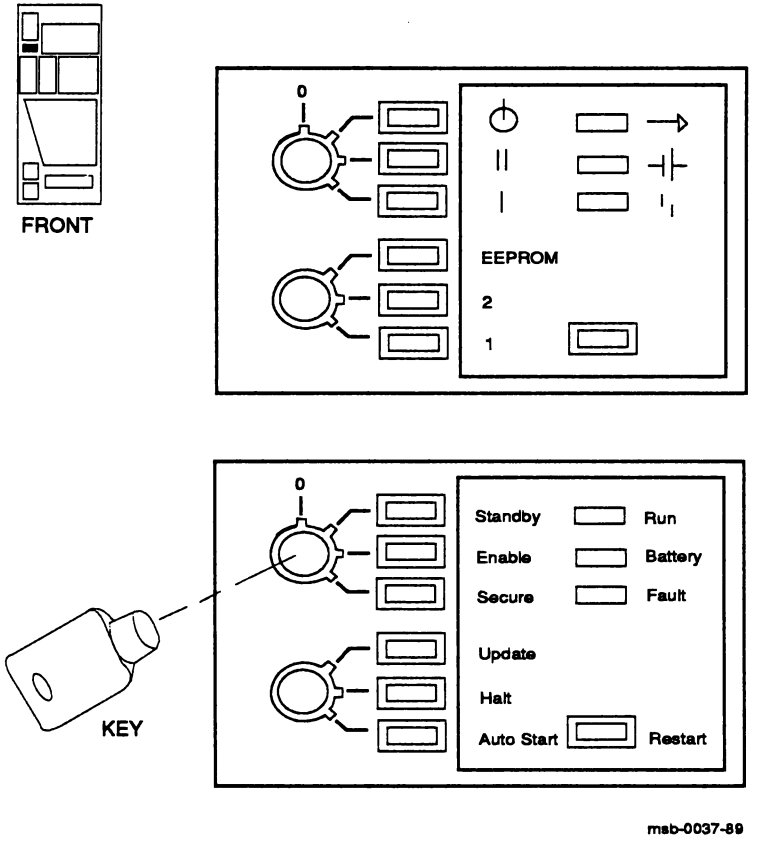


Table 1–1: Upper Key Switch

Position	Effect	Light Color
O (Off)	Removes all power, except to the battery backup unit.	No light
Standby	Supplies power only to memory and blowers.	Red
Enable	Supplies power to whole system; console terminal is enabled. Used for console mode or restart, and to start self-test.	Yellow
Secure (Normal Position)	Prevents entry to console mode; position used while machine is executing programs. Disables Restart button and causes the lower key switch to have the effect of Auto Start, regardless of its setting.	Green

Table 1–2: Lower Key Switch

Position	Effect	Light Color
Update	Enables writing to the EEPROM on the boot processor. Halts boot processor in console mode on power-up or when Restart button is pressed. Used for updating parameters (such as the terminal characteristics and boot specifications) that are stored in each processor's EEPROM (upper key switch must be set to Enable). Prevents an auto restart.	Red
Halt	Prevents an auto restart if a failure or transient power outage occurs.	Yellow
Auto Start (Normal Position)	Allows restart or reboot. Used for normal operation of the system.	Green

Table 1–3: Restart Button

Upper Key Switch	Lower Key Switch	Restart Button Function
Enable	Update or Halt	Runs self-test, then halts.
Enable	Auto Start	Runs self-test, and attempts a restart. If the restart fails, then it reboots the operating system. If the reboot fails, control returns to the console.
Standby or Secure	Any position	Does not function.

Table 1–4: Control Panel Status Indicator Lights

Light	Color	State	Meaning
Run	Green	On	System is executing operating system instructions on at least one processor.
		Off	System is in console mode, is set to Standby, or is turned off.
Battery	Green	On	Battery backup unit is fully charged; normal operation.
		Flashing 1 x/sec	Battery backup unit is charging.
		Flashing 10 x/sec	Battery backup unit is supplying power to the system.
		Off	Either system does not have a battery backup unit or the battery backup unit is turned off.
Fault	Red	On	Self-test is in progress. If light does not turn off, system has a hardware fault. See Chapter 6 for self-test information.
		Off	Self-test has completed, or the system is turned off.

Table 1–5: Console Commands and Qualifiers

Command and Qualifiers	Function
BOOT /R3:n /R5:n /XMI:n /BI:m /NODE:n	Initializes the system, causing a self-test, and begins the boot program.
CLEAR EXCEPTION	Cleans up error state in XBER and RCSR registers.
CONTINUE	Begins processing at the address where processing was interrupted by a CTRL/P console command.
DEPOSIT /B /G /I /L /N /P /V /W	Stores data in a specified address.
EXAMINE /B /G /I /L /N /P /V /W	Displays the contents of a specified address.

Table 1–5 (Cont.): Console Commands and Qualifiers

Command and Qualifiers	Function
FIND /MEMORY /RPB	Searches main memory for a page-aligned 256-Kbyte block of good memory or for a restart parameter block.
HALT	Null command; no action is taken since the processor has already halted in order to enter console mode.
HELP	Prints explanation of console commands.
INITIALIZE [n] /BI:n	Performs a system reset, including self-test.
REPEAT	Executes the command passed as its argument.
RESTORE EEPROM	Copies the TK tape's EEPROM contents to the EEPROM of the processor executing the command.
SAVE EEPROM	Copies to the TK tape the contents of the EEPROM of the processor executing the command.
SET BOOT	Stores a boot command by a nickname.
SET CPU [n] /ENABLED /ALL /NOENABLED /NEXT_PRIMARY /PRIMARY /ALL /NOPRIMARY	Specifies eligibility of processors to become the boot processor.
SET LANGUAGE ENGLISH INTERNATIONAL	Changes the output of the console error messages between numeric code only (international mode) and code plus explanation (English mode).
SET MEMORY /CONSOLE_LIMIT:n /INTERLEAVE:(n+n...) /INTERLEAVE:DEFAULT /INTERLEAVE:NONE	Designates the method of interleaving the memory modules; supersedes the console program's default interleaving.
SET TERMINAL /BREAK /NOBREAK /HARDCOPY /NOHARDCOPY /SCOPE /NOSCOPE /SPEED:n	Sets console terminal characteristics.

Table 1–5 (Cont.): Console Commands and Qualifiers

Command and Qualifiers	Function
SHOW ALL	Displays the current value of parameters set.
SHOW BOOT	Displays all boot commands and nicknames that have been saved using SET BOOT.
SHOW CONFIGURATION	Displays the hardware device type and revision level for each XMI and VAXBI node and indicates self-test status.
SHOW CPU	Identifies the primary processor and the status of other processors.
SHOW ETHERNET	Locates all Ethernet adapters on the system and displays their addresses.
SHOW LANGUAGE	Displays the mode currently set for console error messages, international or English.
SHOW MEMORY	Displays the memory lines from the system self-test, showing interleave and memory size.
SHOW TERMINAL	Displays the baud rate and terminal characteristics functioning on the console terminal.
START	Begins execution of an instruction at the address specified in the command string.
STOP /BI:n	Halts the specified node.
TEST /RBD	Passes control to the self-test diagnostics.
UPDATE	Copies contents of the EEPROM on the processor executing the command to the EEPROM of the processor specified in the command string.
Z /BI:n	Logically connects the console terminal to another processor on the XMI bus or to a VAXBI node.
!	Introduces a comment.

Table 1–6: Console Control Characters

Character	Function
BREAK	Increments the console baud rate if enabled.
CTRL/C	Causes the console to abort processing of a command.
CTRL/O	Causes console to discard output to the console terminal until the next CTRL/O is entered.
CTRL/P	In console mode, acts like CTRL/C. In program mode, causes the boot processor to halt and begin running the console program.
CTRL/Q	Resumes console output that was suspended with CTRL/S.
CTRL/R	Redisplays the current line.
CTRL/S	Suspends console output on the console terminal until CTRL/Q is typed.
CTRL/U	Discards all characters on the current line.
DELETE	Deletes the previously typed character.
ESC	Suppresses any special meaning associated with a given character.
RETURN	Carriage return; ends a command line.

Figure 1–2: BOOT Command Syntax

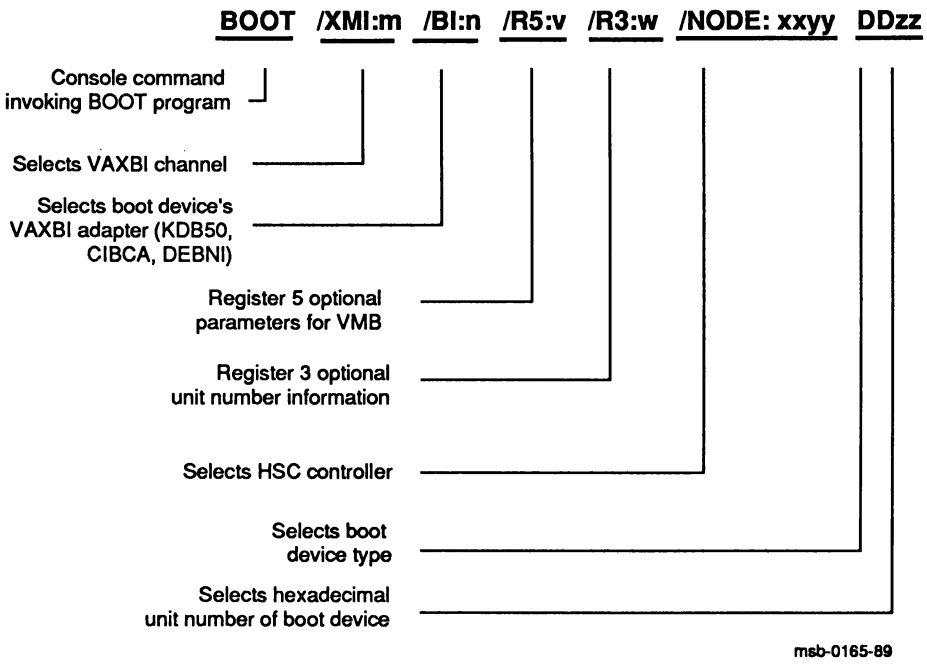


Table 1–7: Sample BOOT Commands

Boot Procedure	BOOT command
Boot from TK tape drive	BOOT CSA1
Boot from local disk	BOOT /XMI:m /BI:n DUzz
Boot from HSC disk	BOOT /XMI:m /BI:n /R5:v /NODE:xxyy DUzz
Boot VAX/DS from TK	BOOT /R5:10 CSA1
Boot VAX/DS from disk	BOOT /XMI:m /BI:n /R5:10 DUzz
Conversational boot	BOOT /XMI:m /BI:n /R5:1 DDzz
Boot from VMS shadow set	BOOT /XMI:m /BI:n /R3:w /NODE:xxyy DDzz
Boot over the Ethernet	BOOT /XMI:m /BI:n ET0

Table 1–8: BOOT Command Qualifiers

Qualifier	Function
/X[MI]:number	Specifies the XMI node number of the node that connects the boot device.
/B[I]:number	Specifies the VAXBI node that connects the boot device. The /XMI qualifier must have selected a node containing a DWMBA/A.
/R5:number	Specifies the hexadecimal value to be loaded into register R5 immediately before the virtual memory boot (VMB) program receives control. Use as a bit mask to select VMB options and to set the system root directory.
/R3:number	Specifies the hexadecimal value to be loaded into register R3 immediately before the virtual memory boot (VMB) program receives control. This qualifier is used when multiple unit numbers must be specified: for example, when booting from VMS shadow sets. If /R3 is specified, the unit number portion of the device name is ignored.
/N[ODE]:number	Specifies the remote node(s) that provide access to the boot device. The /XMI (and optionally /BI) qualifiers must have identified a controller that supports "nodes" such as a VAXcluster adapter. The /NODE qualifier would then specify the VAXcluster node number(s) of the HSC controlling the boot device.

Table 1–9: R5 Bit Functions for VMS

Bit	Function
0	Conversational boot. The secondary bootstrap program, SYSBOOT, prompts you for system parameters at the console terminal.
1	Debug. If this flag bit is set, the operating system maps the code for the XDELTA debugger into the system page tables of the running operating system.
2	Initial breakpoint. If this flag bit is set, VMS executes a breakpoint (BPT) instruction early in the bootstrapping process.
3	Secondary boot from boot block. The secondary boot is a single 512-byte block whose logical block number is specified in General Purpose Register R4.
4	Boots the VAX Diagnostic Supervisor. The secondary loader is an image called DIAGBOOT.EXE.
5	Boot breakpoint. This stops the primary and secondary loaders with a breakpoint (BPT) instruction before testing memory.
6	Image header. The transfer address of the secondary loader image comes from the image header for that file. If this flag is not set, control shifts to the first byte of the secondary loader.
8	File name. VMB prompts for the name of a secondary loader.
9	Halt before transfer. VMB executes a HALT instruction before transferring control to the secondary loader.
13	No effect, since console program tests memory.
15	Reserved for VAX Diagnostic Supervisor.
16	Do not discard CRD pages.
31:28	Specifies the top-level directory number for system disks.

Table 1–10: R5 Bit Functions for ULTRIX

Bit	Function
0	Forces ULTRIXBOOT to prompt the user for an image name (the default is VMUNIX).
1	Boots the ULTRIX kernel image in single-user mode.
3	Must be set, and R4 must be zero.
16	Must be set.

Table 1–11: Console Error Messages Indicating Halt

Error Message	Meaning
?02 External halt (CTRL/P, break, or external halt).	CTRL/P or STOP command.
?03 Power-up halt.	System has powered up, had a system reset, or an XMI node reset.
?04 Interrupt stack not valid during exception processing.	Interrupt stack pointer contained an invalid address.
?05 Machine check occurred during exception processing.	A machine check occurred while handling another error condition.
?06 Halt instruction executed in kernel mode.	The CPU executed a Halt instruction.
?07 SCB vector bits <1:0> = 11.	An interrupt or exception vector in the System Control Block contained an invalid address.
?08 SCB vector bits <1:0> = 10.	An interrupt or exception vector in the System Control Block contained an invalid address.
?0A CHMx executed while on interrupt stack.	A change-mode instruction was issued while executing on the interrupt stack.
?10 ACV/TNV occurred during machine check processing.	An access violation or translation-not-valid error occurred while handling another error condition.
?11 ACV/TNV occurred during kernel-stack-not-valid processing.	An access violation or translation-not-valid error occurred while handling another error condition.
?12 Machine check occurred during machine check processing.	A machine check occurred while processing a machine check.
?13 Machine check occurred during kernel-stack-not-valid processing.	A machine check occurred while handling another error condition.
?19 PSL <26:24>= 101 during interrupt or exception.	An exception or interrupt occurred while on the interrupt stack but not in kernel mode.
?1A PSL <26:24>= 110 during interrupt or exception.	An exception or interrupt occurred while on the interrupt stack but not in kernel mode.
?1B PSL <26:24>= 111 during interrupt or exception.	An exception or interrupt occurred while on the interrupt stack but not in kernel mode.

Table 1–11 (Cont.): Console Error Messages Indicating Halt

Error Message	Meaning
?1D PSL <26:24> = 101 during REI.	An REI instruction attempted to restore a PSL with an invalid combination of access mode and interrupt stack bits.
?1E PSL <26:24> = 110 during REI.	An REI instruction attempted to restore a PSL with an invalid combination of access mode and interrupt stack bits.
?1F PSL <26:24> = 111 during REI.	An REI instruction attempted to restore a PSL with an invalid combination of access mode and interrupt stack bits.

Table 1–12: Standard Console Error Messages

Error Message	Meaning
?20 Illegal memory reference.	An attempt was made to reference a virtual address (/V) that is either unmapped or is protected against access under the current PSL.
?21 Illegal command.	The command was not recognized, contained the wrong number of parameters, or contained unrecognized or inappropriate qualifiers.
?22 Illegal address.	The specified address was recognized as being invalid, for example, a general purpose register number greater than 15.
?23 Value is too large.	A parameter or qualifier value contained too many digits.
?24 Conflicting qualifiers.	A command specified recognized qualifiers that are illegal in combination.
?25 Checksum did not match.	The checksum calculated for a block of X command data did not match the checksum received.
?26 Halted.	The processor is currently halted.
?27 Item was not found.	The item requested in a FIND command could not be found.

Table 1–12 (Cont.): Standard Console Error Messages

Error Message	Meaning
?28 Timeout while waiting for characters.	The X command failed to receive a full block of data within the timeout period.
?29 Machine check accessing memory.	Either the specified address is not implemented by any hardware in the system, or an attempt was made to write a read-only address, for example, the address of the 33rd Mbyte of memory on a 32-Mbyte system.
?2A Unexpected machine check or interrupt.	A valid operation within the console caused a machine check or interrupt.
?2B Command is not implemented.	The command is not implemented by this console.
?2C Unexpected exception.	An attempt was made to examine either a nonexistent IPR or an unimplemented register in RSSC address range (20140000—20140800).
?2D For Secondary Processor n.	This message is a preface to second message describing some error related to a secondary processor. This message indicates which secondary processor is involved.
?2E Specified node is not an I/O adapter.	The referenced node is incapable of performing I/O or did not pass its self-test.
?30 Write to Z command target has timed out.	The target node of the Z command is not responding.
?31 Z connection terminated by ^P.	A CTRL/P was typed on the keyboard to terminate a Z command.
?32 Your node is already part of a Z connection.	You cannot issue a Z command while executing a Z command.
?33 Z connection successfully started.	You have requested a Z connection to a valid node.
?34 Specified target already has a Z connection.	The target node was the target of a previous Z connection that was improperly terminated. Reset the system to clear this condition.
?36 Command too long.	The command length exceeds 80 characters.

Table 1–12 (Cont.): Standard Console Error Messages

Error Message	Meaning
?37 Explicit interleave list is bad. Configuring all arrays uninterleaved.	The list of memory arrays for explicit interleave includes no nodes that are actually memory arrays. All arrays found in the system are configured.
?39 Console patches are not usable.	The console patch area in EEPROM is corrupted or contains a patch revision that is incompatible with the console ROM.
?3B Error encountered during I/O operation.	An I/O adapter returned an error status while the console boot primitive was performing I/O.
?3C Secondary processor not in console mode.	The primary processor console needed to communicate with a secondary processor, but the secondary processor was not in console mode. STOP the node or reset the system to clear this condition.
?3D Error initializing I/O device.	A console boot primitive needed to perform I/O, but could not initialize the I/O adapter.
?3E Timeout while sending message to secondary processor.	A secondary processor failed to respond to a message sent from the primary. The primary sends such messages to perform console functions on secondary processors.
?3F Microcode power-up self-test failed in REX520.	CPU chip failed its microcoded self-test.
?40 Key switch must be at "Update" to update EEPROM.	A SET command was issued, but the key switch was not set to allow updates to the EEPROM.
?41 Specified node is not a bus adapter.	A command to access a VAXBI node specified an XMI node that was not a bus adapter.
?42 Invalid terminal speed.	The SET TERMINAL command specified an unsupported baud rate.
?43 Unable to initialize node.	The INITIALIZE command failed to reset the specified node.

Table 1–12 (Cont.): Standard Console Error Messages

Error Message	Meaning
?44 Processor is not enabled to BOOT or START.	As a result of a SET CPU/NOENABLE command, the processor is disabled from leaving console mode.
?45 Unable to stop node.	The STOP command failed to halt the specified node.
?46 Memory interleave set is inconsistent: n n ...	The listed nodes do not form a valid memory interleave set. One or more of the nodes might not be a memory array or might be of a different size, or the set could contain an invalid number of members. Each listed array that is a valid memory will be configured uninterleaved.
?47 Insufficient working memory for normal operation.	Less than 256 Kbytes per processor of working memory were found. There is insufficient memory for the console to function normally or for the operating system to boot.
?48 Uncorrectable memory errors—long memory test must be performed.	A memory array contains an unrecoverable error. The console must perform a slow test to locate all the failing locations.
?49 Memory cannot be initialized.	The specified operation was attempted and prevented.
?4A Memories not interleaved due to uncorrectable errors:	The listed arrays would normally have been interleaved (by default or explicit request). Because one or more of them contained unrecoverable errors, this interleave set will not be constructed.
?4B Internal logic error in console.	The console encountered a theoretically impossible condition.
?4C Invalid node for Z command.	The target of a Z command must be a CPU or an I/O adapter and must not be the primary processor.
?4D Invalid node for new primary.	The SET CPU command failed when attempting to make the specified node the primary processor.
?4E Specified node is not a processor.	The specified node is not a processor, as required by the command.

Table 1–12 (Cont.): Standard Console Error Messages

Error Message	Meaning
?4F System serial number has not been initialized.	No CPU in the system contains a valid system serial number.
?50 System serial number not initialized on primary processor.	The primary processor has an uninitialized system serial number. All other processors in the system contain a valid serial number.
?51 Secondary processor returned bad response message.	A secondary processor returned an unintelligible response to a request made by the console on the primary processor.
?52 ROM revision mismatch. Secondary processor has revision *x.xx*.	The revision of console ROM of a secondary processor does not match that of the primary.
?53 EEPROM header is corrupted.	The EEPROM header has been corrupted. The EEPROM must be restored from the TK tape drive.
?54 EEPROM revision mismatch. Secondary processor has revision *x.xx/y.yy*.	A secondary processor has a different revision of EEPROM or has a different set of EEPROM patches installed.
?55 Failed to locate EEPROM area.	The EEPROM did not contain a set of data required by the console. The EEPROM may be corrupted.
?56 Console parameters on secondary processor do not match primary.	The console parameters are not the same for all processors .
?57 EEPROM area checksum error.	A portion of the EEPROM is corrupted. It may be necessary to reload the EEPROM from the TK tape drive.
?58 Saved boot specifications on secondary processor do not match primary.	The saved boot specifications are not the same for all processors.
?59 Invalid unit number.	A BOOT or SET BOOT command specified a unit number that is not a valid hexadecimal number between 0 and FF.
?5A System serial number mismatch. Secondary processor has xxxxxxxx.	The indicated serial number of a secondary processor does not match that of the primary.

Table 1–12 (Cont.): Standard Console Error Messages

Error Message	Meaning
?5B Unknown type of boot device.	The console program does not have a boot primitive to support the specified type of device or the device could not be accessed to determine its type.
?5C No HELP is available.	The HELP command is not supported when the console language is set to International.
?5D No such boot spec found.	The specified boot specification was not found in the EEPROM.
?5E Saved boot spec table full.	The maximum number of saved boot specifications has already been stored.
?5F EEPROM header version mismatch.	Processors have different versions of EEPROMs.
?61 EEPROM header or area has bad format.	All or part of the EEPROM contains inconsistent data and is probably corrupted. Reload the EEPROM from the TK tape.
?62 Illegal node number.	The specified node number is invalid.
?63 Unable to locate console tape device.	The console could not locate the I/O adapter that controls the TK tape.
?64 Operation only applies to secondary processors.	The command can only be directed at a secondary processor.
?65 Operation not allowed from secondary processor.	A secondary processor cannot perform this operation.
?66 Validation of EEPROM tape image failed.	The image on tape is corrupted or is not the result of a SAVE EEPROM command. The image cannot be restored.
?67 Read of EEPROM image from tape failed.	The EEPROM image was not successfully read from tape.
?68 Validation of local EEPROM failed.	For a PATCH EEPROM operation, the EEPROM must first contain a valid image before it can be patched. For a RESTORE EEPROM operation, the image was written back to EEPROM but could not be read back successfully.
?69 EEPROM not changed.	The EEPROM contents were not changed.

Table 1–12 (Cont.): Standard Console Error Messages

Error Message	Meaning
?6A EEPROM changed successfully.	The EEPROM contents were successfully patched or restored.
?6B Error changing EEPROM.	An error occurred in writing to the EEPROM. The EEPROM contents may be corrupted.
?6C EEPROM saved to tape successfully.	The EEPROM contents were successfully written to the TK tape.
?6D EEPROM not saved to tape.	The EEPROM contents were not completely written to the TK tape.
?6E EEPROM Revision = *x.xx*/*y.yy*.	The EEPROM contents are at revision *x.xx* with revision *y.yy* patches.
?6F Major revision mismatch between tape image and EEPROM.	The major revision of tape and EEPROM do not match. The requested operation cannot be performed.
?70 Tape image Revision = *x.xx*/*y.yy*.	The EEPROM image on the TK tape is at revision *x.xx* with revision *y.yy* patches.
?73 System serial number updated.	The EEPROM has been updated with the correct system serial number.
?74 System serial number not updated.	The EEPROM has not been changed.
?75 /CONSOLE_LIMIT value too small for proper operation. Value ignored.	No change has been made.
?76 Error writing to tape. Tape may be write-locked.	Tape has not been written. Check to see if tape is write-locked.
?77 CCA not accessible or corrupted.	Attempt to find the console communications area (CCA) failed. The console then builds a local CCA, which does not allow for interprocessor communication.
?78 CONTINUE command is disabled, because DC520 revision is less than 3.	Systems with pass 1 and pass 2 DC520 chips do not implement the CONTINUE console command.
?83 Loading system software.[1]	The console is attempting to load the operating system in response to a BOOT command, power-up, or restart failure.

[1]No numbered prefix appears with these messages in English language mode. These numbers are used for these messages in International mode.

Table 1–12 (Cont.): Standard Console Error Messages

Error Message	Meaning
?84 Failure.[1]	An operation did not complete successfully. Should be issued with another message to clarify failure.
?85 Restarting system software.[1]	The console is attempting to restart the in-memory copy of the operating system following a power-up or serious error.
?A0 Initializing system.[1]	The console is resetting the system in response to a BOOT command.
?A6 Console halting after unexpected machine check or exception.[1]	The console executed a Halt instruction to reset the console state after processing an unexpected machine check.
?A7 RCSR <WD> is set. Local CCA must be built.	When the <WD> bit is set, writes to memory are disabled. The processor must then build a CCA in local memory. Main memory cannot be written to or accessed with interlocked instructions.
?A8 Bootstrap failed due to previous error.[1]	The previous attempt to bootstrap the system failed.
?A9 Restart failed due to previous error.[1]	The previous attempt to restart the system failed.
Node: n ?xx	Error message ?xx was generated on secondary processor *n* and was passed to the primary processor to be displayed.

[1]No numbered prefix appears with these messages in English language mode. These numbers are used for these messages in International mode.

Chapter 2
Self-Test

Example 2–1 is a sample self-test display, which deliberately includes some failures to illustrate the type of information reported. Each line is described below. Table 2–1 describes the configuration and assumptions used for this sample.

Example 2–1: Sample Self-Test Results

```
#123456789 0123456789 0123456789 01234567#❶

F   E   D   C   B   A   9   8   7   6   5   4   3   2   1   0   NODE # ❷

    A   A   .   .   M   M   M   M   .   .   P   P   P   P       TYP    ❸
    o   o   .   .   +   +   +   +   .   .   +   +   -   +       STF    ❹
    .   .   .   .   .   .   .   .   .   .   E   B   E   D       BPD    ❺
    .   .   .   .   .   .   .   .   .   .   +   -   -   +       ETF    ❻
    .   .   .   .   .   .   .   .   .   .   B   E   E   D       BPD    ❼

.   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   XBI D -❽
.   .   .   .   .   .   .   .   .   +   +   +   .   -   +   .   XBI E +

    .   .   .   .  B2   B1  A2  A1   .   .   .   .   .   .       ILV    ❾
    .   .   .   .  32   32  32  32   .   .   .   .   .   .       128Mb  ❿

ROM0 = V1.00  ROM1 = V1.00⓫  EEPROM = 1.00/1.01⓬  SN = SG01234567⓭
>>>
```

❶ The progress trace. This line appears when slot 1 holds a KA64A module.

❷ Identifies the node number (NODE #).

Lines 3 through 7 refer to XMI node numbers; the XBI lines refer to VAXBI node numbers.

❸ Identifies the module type (TYP).

P = processor
M = memory
A = adapter

❹ Gives self-test failure results (STF).

+ = passed
– = failed
o = not tested as part of the initial power-up test

❺ Shows boot processor designation (BPD).

E = eligible to be boot processor
D = ineligible to be boot processor
B = designated as boot processor

❻ Gives extended CPU/memory tests failure results (ETF). Same interpretation as STF.

❼ Shows the second boot processor designation, which may be different from that on the first BPD line.

❽ Shows DWMBA test results, node number, and self-test results of the VAXBI nodes (XBI). The + or – at the right means that the DWMBA passed or failed when tested by the boot processor. If the DWMBA passed, a + or – corresponding to each VAXBI node indicates whether that node passed or failed its own self-test.

❾ Displays the memory array membership in interleave sets (ILV). Each letter denotes a different interleave set.

❿ Gives each memory array size and the total working memory size (Mb).

⓫ Shows the version number of the boot processor's ROMs (ROM0 and ROM1).

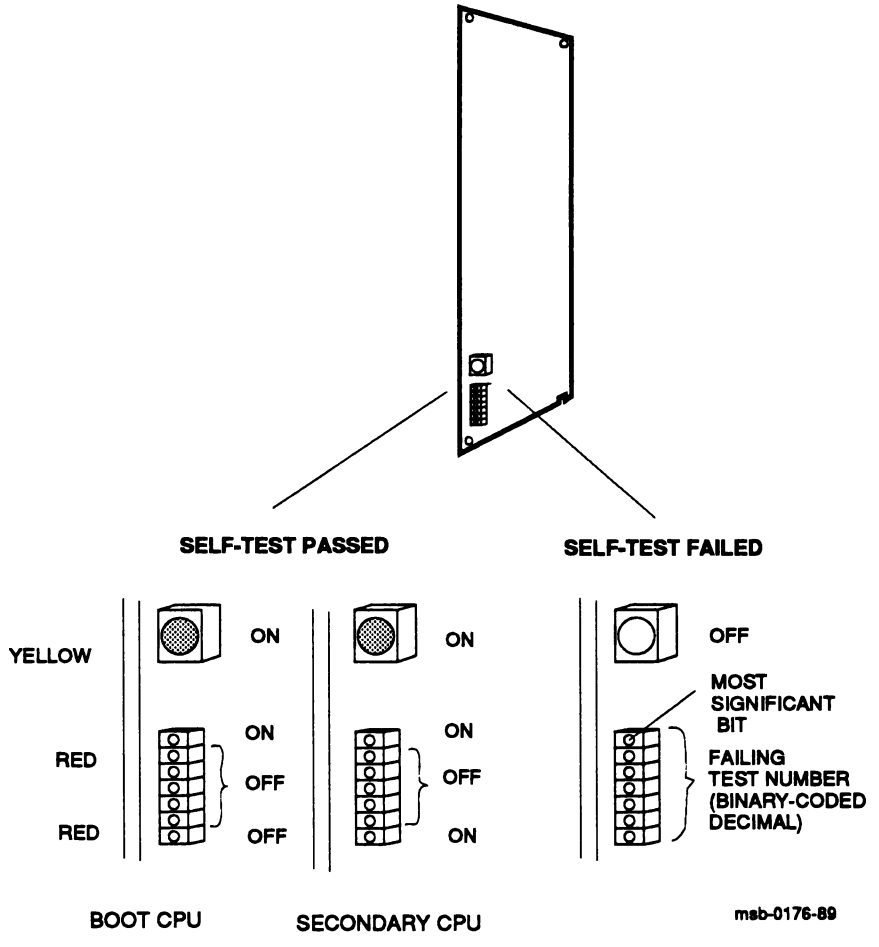
⓬ Gives the version number and revision number of the boot processor's EEPROM. The first number is the base revision of the EEPROM, which rarely changes. The second number is the revision of console and diagnostic patches applied to the EEPROM. This number increments with every patch operation. For example, revision 1.01 means that there has been one patch for base revision 1.00.

⓭ Lists the serial number of the system (SN).

Table 2–1: System Configuration for Sample Self-Test

Module	XMI Node Number	Module Type
KA64A	1	Processor; disabled from being boot processor.
KA64A	2	Processor; fails self-test.
KA64A	3	Processor; boot processor after self-test, fails extended test.
KA64A	4	Processor; becomes boot processor.
MS62A	7	Memory (32 Mbytes); interleaved with memory at node 8.
MS62A	8	Memory (32 Mbytes); interleaved with memory at node 7.
MS62A	9	Memory (32 Mbytes); interleaved with memory at node A.
MS62A	A	Memory (32 Mbytes); interleaved with memory at node 9.
DWMBA/A	D	I/O adapter that failed self-test.
DWMBA/A	E	I/O adapter leading to a VAXBI bus that has passing modules at nodes 1, 4, 5, and 6; node 2 on the VAXBI bus failed self-test.

Figure 2–1: KA64A LEDs After Self-Test



NOTE: *Interpretation of small red LEDs: ON is a zero, and OFF is a one.*

Table 2–2: TK70 Light Summary

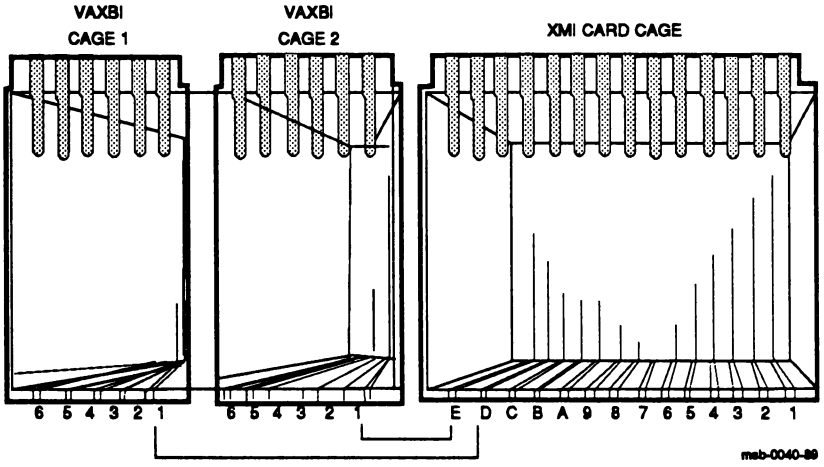
Light	State	Condition
Green (Operate Handle)	On	OK to operate handle.
	Off	Do not operate handle.
	Blinking	Defective cartridge. Pull the handle to the open position and remove cartridge. Try another cartridge.
Yellow (Tape in Use)	Steady	Drive ready.
	Blinking	Drive in use.
Orange [1] (Write Protected)	On	Tape write protected.
	Off	Tape write enabled.
All three lights	Blinking	Drive fault. Attempt to reset the fault by pressing the unload button.

[1]The orange light is on when any of the following conditions exist:
- Cartridge write protect switch is in the protected position.
- Cartridge is software write protected.
- Attempt was made to mount or initialize a cartridge previously written in a TK50 drive.

Chapter 3
Address Space

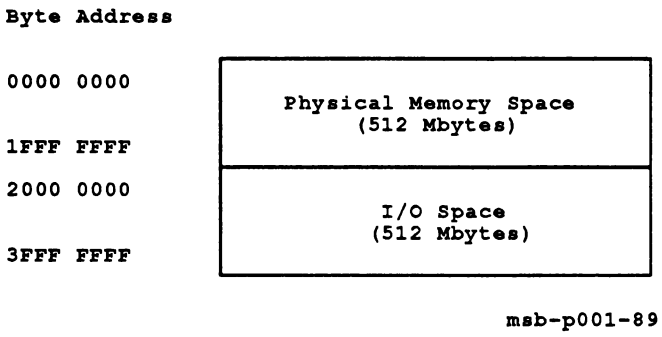
The design of the hardware for the system bus (the XMI) and for the VAXBI bus affects addressing. The VAXBI card cage was designed so that node addressing did not depend upon a particular slot; a node's address was derived from a node ID plug in the backplane, a number from 0 to 15. VAXBI address space was divided up to provide for 16 nodes. The system bus card cage, on the other hand, has its 14 slots permanently assigned to specific address locations. Furthermore, because of the XMI architecture and the XMI cage design, no modules that require I/O cables can be installed in the middle six slots (slots 5 through A).

Figure 3–1: VAX 6000–400 Slot Numbers



The system bus address space is equally divided between memory and I/O space (see Figure 3–2). Figure 3–3 shows how the XMI I/O address space is allocated. Each node on the XMI has its own nodespace of 512 Kbytes. Sixteen 512-Kbyte blocks are in XMI nodespace, but the system bus implements only 14 nodes, leaving the first and last blocks reserved. The 192 reserved Mbytes shown in the center of the figure correspond to the slots that cannot be used for I/O. Eight blocks are assigned to I/O adapters.

Figure 3–2: XMI Memory and I/O Address Space



Register addresses for a particular device in a system are found by adding an offset to the base address for that device. To distinguish between addresses in XMI address space and addresses in VAXBI address space, we use the following convention:

lowercase bb + offset indicates an address in VAXBI address space
uppercase BB + offset indicates an address in XMI address space

Figure 3–3: XMI I/O Space Address Allocation

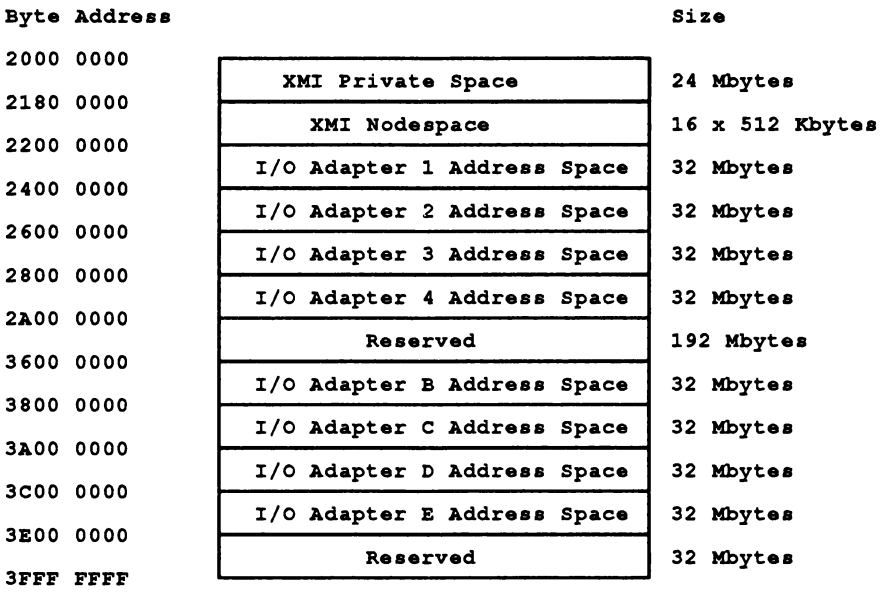


Table 3–1: XMI Addressing

Slot	Node	Permissible Modules[1]	Nodespace Base Address (BB)	I/O Window Space Prefix
1	1	CPU, I/O	2188 0000	22xx xxxx
2	2	CPU, Mem, I/O	2190 0000	24xx xxxx
3	3	CPU, Mem, I/O	2198 0000	26xx xxxx
4	4	CPU, Mem, I/O	21A0 0000	28xx xxxx
5	5	CPU, Mem	21A8 0000	N/A
6	6	CPU, Mem	21B0 0000	N/A
7	7	CPU, Mem	21B8 0000	N/A
8	8	CPU, Mem	21C0 0000	N/A
9	9	CPU, Mem	21C8 0000	N/A
10	A	CPU, Mem	21D0 0000	N/A
11	B	CPU, Mem, I/O	21D8 0000	36xx xxxx
12	C	CPU, Mem, I/O	21E0 0000	38xx xxxx
13	D	CPU, Mem, I/O	21E8 0000	3Axx xxxx
14	E	CPU, I/O	21F0 0000	3Cxx xxxx

[1]Key to permissible modules:

CPU = KA64A CPU module
Mem = MS62A memory module
I/O = DWMBA/A adapter module

3.1 How to Find a Register in XMI Address Space

Because XMI addresses correspond to slot and node numbers, you want to determine the slot of the XMI card cage in which the module resides. The slot number can be determined in two ways:

- By looking at the XMI card cage (numbering of slots is shown in Figure 3–1)
- By entering at the console a SHOW CONFIGURATION command

A typical response is shown below.

```
>>> SHOW CONFIGURATION

      Type            Rev
  1+  KA64A    (8082) 0006
  2+  KA64A    (8082) 0006
  9+  MS62A    (4001) 0002
  A+  MS62A    (4001) 0002
  D+  DWMBA/A  (2001) 0002
  E+  DWMBA/A  (2001) 0002

  XBI D
  1+  DWMBA/B  (2107) 0007
  5+  DMB32    (0109) 210B
  6+  DEBNI    (0118) 0100

  XBI E
  1+  DWMBA/B  (2107) 0007
  4+  KDB50    (010E) 0F1C
  6+  TBK70    (410B) 0307
```

Assume that you want to examine the Bus Error Register (XBER) of the DWMBA/A module in the leftmost slot (14), which is XMI node E. From Table 3–1 you can see that the nodespace base address for the XMI module at node E is 21F0 0000. From Table 6–1 you can see that the XBER offset is BB + 04, so you add 04 to the base address to get the address for that module's XBER register. You could examine the XBER register with the command:

```
>>>  E/L/P 21F00004
```

3.2 How to Find a Register in VAXBI Address Space

The first part of a VAXBI adapter's physical XMI address depends on which XMI slot the DWMBA/A module occupies. The second part of the address depends on the adapter's VAXBI node number, which is shown in the SHOW CONFIGURATION display.

NOTE: *VAXBI slot and node numbers are not identical. The placement of the VAXBI node ID plug on the backplane determines the node ID, so seeing that a particular option is in a certain slot does not guarantee that the slot and node number are identical. Use the VAXBI node identification from the SHOW CONFIGURATION command.*

The XMI slot number can be determined in two ways:

- By looking at the XMI card cage (numbering of slots is shown in Figure 3–1)
- By entering at the console a SHOW CONFIGURATION command

A typical response is shown below.

```
>>> SHOW CONFIGURATION

      Type             Rev
  1+  KA64A    (8082)  0006
  2+  KA64A    (8082)  0006
  9+  MS62A    (4001)  0002
  A+  MS62A    (4001)  0002
  D+  DWMBA/A  (2001)  0002
  E+  DWMBA/A  (2001)  0002

  XBI D
  1+  DWMBA/B  (2107)  0007
  5+  DMB32    (0109)  210B
  6+  DEBNI    (0118)  0100

  XBI E
  1+  DWMBA/B  (2107)  0007
  4+  KDB50    (010E)  0F1C
  6+  TBK70    (410B)  0307
```

Assume that you want to examine the Device Register (DTYPE) for the DEBNI, which is node 6 in the first VAXBI channel shown above (XBI D). Figure 3–1 also shows which VAXBI cage is linked to which XMI slot.

To get the address for the DEBNI Device Register (DTYPE), do the following:

1. From Table 3–1 find XMI node D and take the 2-digit prefix for that node's window space (3A).

2. From Table 3–2 find VAXBI node 6 and in column 2 you can see that the starting address for VAXBI node 6 is xx00 C000.
3. Combine this second number with the 2-digit prefix. You now have the adapter's base address (3A00 C000) in VAXBI address space, indicated by lowercase bb.
4. From Table 3–3, VAXBI Registers, you can see that the VAXBI Device Register (DTYPE) is at bb + 00, which is 3A00 C000.

The Device Register for the DEBNI would be examined by:

```
>>> E/L/P 3A00C000
```

Table 3–2: VAXBI Nodespace and Window Space Address Assignments

Node	Nodespace Addresses		Window Space Addresses	
Number	Starting	Ending	Starting	Ending
0	xx00 0000	xx00 1FFF	xx40 0000	xx43 FFFF
1	xx00 2000	xx00 3FFF	xx44 0000	xx47 FFFF
2	xx00 4000	xx00 5FFF	xx48 0000	xx4B FFFF
3	xx00 6000	xx00 7FFF	xx4C 0000	xx4F FFFF
4	xx00 8000	xx00 9FFF	xx50 0000	xx53 FFFF
5	xx00 A000	xx00 BFFF	xx54 0000	xx57 FFFF
6	xx00 C000	xx00 DFFF	xx58 0000	xx5B FFFF
7	xx00 E000	xx00 FFFF	xx5C 0000	xx5F FFFF
8	xx01 0000	xx01 1FFF	xx60 0000	xx63 FFFF
9	xx01 2000	xx01 3FFF	xx64 0000	xx67 FFFF
A	xx01 4000	xx01 5FFF	xx68 0000	xx6B FFFF
B	xx01 6000	xx01 7FFF	xx6C 0000	xx6F FFFF
C	xx01 8000	xx01 9FFF	xx70 0000	xx73 FFFF
D	xx01 A000	xx01 BFFF	xx74 0000	xx77 FFFF
E	xx01 C000	xx01 DFFF	xx78 0000	xx7B FFFF
F	xx01 E000	xx01 FFFF	xx7C 0000	xx7F FFFF

Table 3–3: VAXBI Registers

Name	Mnemonic	Address[1]
Device Register	DTYPE	bb+00
VAXBI Control and Status Register	VAXBICSR	bb+04
Bus Error Register	BER	bb+08
Error Interrupt Control Register	EINTRSCR	bb+0C
Interrupt Destination Register	INTRDES	bb+10
IPINTR Mask Register	IPINTRMSK	bb+14
Force-Bit IPINTR/STOP Destination Register	FIPSDES	bb+18
IPINTR Source Register	IPINTRSRC	bb+1C
Starting Address Register	SADR	bb+20
Ending Address Register	EADR	bb+24
BCI Control and Status Register	BCICSR	bb+28
Write Status Register	WSTAT	bb+2C
Force-Bit IPINTR/STOP Command Register	FIPSCMD	bb+30
User Interface Interrupt Control Register	UINTRCSR	bb+40
General Purpose Register 0	GPR0	bb+F0
General Purpose Register 1	GPR1	bb+F4
General Purpose Register 2	GPR2	bb+F8
General Purpose Register 3	GPR3	bb+FC
Slave-Only Status Register	SOSR	bb+100
Receive Console Data Register	RXCD	bb+200

[1]The abbreviation "bb" refers to the base address of a VAXBI node (the address of the first location of the nodespace).

Chapter 4
KA64A CPU Module Registers

The KA64A registers consist of the following:

- Internal processor registers (IPRs) (see Table 4–2)
- Registers in XMI private space (see Table 4–3)
- XMI registers (see Table 4–4)

Machine-check parameters are listed in Section 4.4 and parse trees in Section 4.5.

Table 4–1: Types of Registers and Bits

Type	Description
RO	Read only
R/W	Read/write
R/W1C	Read/cleared by writing a one
WO	Write only

4.1 KA64A Internal Processor Registers

Table 4–2: KA64A Internal Processor Registers

Register	Mnemonic	Address decimal (hex)	Type	Class
Kernel Stack Pointer	KSP	0 (0)	R/W	1
Executive Stack Pointer	ESP	1 (1)	R/W	1
Supervisor Stack Pointer	SSP	2 (2)	R/W	1
User Stack Pointer	USP	3 (3)	R/W	1
Interrupt Stack Pointer	ISP	4 (4)	R/W	1
Reserved		5–7 (5–7)		3
P0 Base	P0BR	8 (8)	R/W	1
P0 Length	P0LR	9 (9)	R/W	1
P1 Base	P1BR	10 (A)	R/W	1
P1 Length	P1LR	11 (B)	R/W	1
System Base	SBR	12 (C)	R/W	1
System Length	SLR	13 (D)	R/W	1
Reserved		14–15 (E–F)		3
Process Control Block Base	PCBB	16 (10)	R/W	1
System Control Block Base	SCBB	17 (11)	R/W	1

Key to Types:

R–Read
W–Write
R/W–Read/write

Key to Classes:

1–Implemented by the KA64A (as specified in the *VAX Architecture Reference Manual*).
2–Implemented uniquely by the KA64A.
3–Not implemented. Read as zero; NOP on write.
4–Access not allowed; accesses result in a reserved operand fault.
5–Accessible, but not fully implemented; accesses yield UNPREDICTABLE results.
I–The register is initialized on KA64A reset (power-up, system reset, and node reset).

Table 4–2 (Cont.): KA64A Internal Processor Registers

Register	Mnemonic	Address decimal (hex)	Type	Class
Interrupt Priority Level	IPL	18 (12)	R/W	1 I
AST Level	ASTLVL	19 (13)	R/W	1 I
Software Interrupt Request	SIRR	20 (14)	W	1
Software Interrupt Summary	SISR	21 (15)	R/W	1 I
Reserved		22–23 (16–17)		3
Interval Counter Control and Status	ICCS	24 (18)	R/W	2 I
Reserved		25–26 (19–1A)		3
Time-of-Year Clock	TODR	27 (1B)	R/W	1
Console Storage Receiver Status	CSRS	28 (1C)	R/W	5 I
Console Storage Receiver Data	CSRD	29 (1D)	R	5 I
Console Storage Transmitter Status	CSTS	30 (1E)	R/W	5 I
Console Storage Transmitter Data	CSTD	31 (1F)	W	5 I
Console Receiver Control/Status	RXCS	32 (20)	R/W	2 I
Console Receiver Data Buffer	RXDB	33 (21)	R	2 I
Console Transmitter Control/Status	TXCS	34 (22)	R/W	2 I
Console Transmitter Data Buffer	TXDB	35 (23)	W	2 I
Reserved		36–37 (24–25)		3
Machine Check Error Summary	MCESR	38 (26)	W	2
Reserved		39 (27)		3
Accelerator Control and Status	ACCS	40 (28)	R/W	2 I
Reserved		41 (29)		3
Console Saved PC	SAVPC	42 (2A)	R	2
Console Saved PSL	SAVPSL	43 (2B)	R	2
Reserved		44–46 (2C–2E)		3
Translation Buffer Tag	TBTAG	47 (2F)	W	2
Reserved		48–54 (30–36)		3

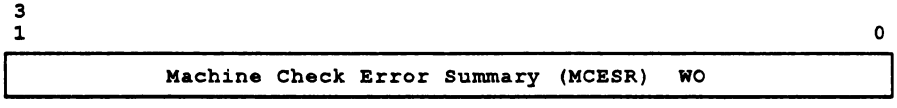
Table 4–2 (Cont.): KA64A Internal Processor Registers

Register	Mnemonic	Address decimal (hex)	Type	Class
I/O Reset	IORESET	55 (37)	W	2
Memory Management Enable	MAPEN	56 (38)	R/W	1 I
Translation Buffer Invalidate All	TBIA	57 (39)	W	1
Translation Buffer Invalidate Single	TBIS	58 (3A)	W	1
Translation Buffer Data	TBDATA	59 (3B)	W	2
Reserved		60–61 (3C–3D)		3
System Identification	SID	62 (3E)	R	1
Translation Buffer Check	TBCHK	63 (3F)	W	1
Reserved		64–111 (40–6F)		3
Backup Cache Reserved	BC112	112 (70)	R/W	5
Backup Cache Backup Tag Store	BCBTS	113 (71)	R/W	2
Backup Cache P1 Tag Store	BCP1TS	114 (72)	R/W	2
Backup Cache P2 Tag Store	BCP2TS	115 (73)	R/W	2
Backup Cache Refresh	BCRFR	116 (74)	R/W	2
Backup Cache Index	BCIDX	117 (75)	R/W	2
Backup Cache Status	BCSTS	118 (76)	R/W	2 I
Backup Cache Control	BCCTL	119 (77)	R/W	2 I
Backup Cache Error	BCERR	120 (78)	R	2
Backup Cache Flush Backup Tag Store	BCFBTS	121 (79)	W	2
Backup Cache Flush Primary Tag Store	BCFPTS	122 (7A)	W	2
Reserved		123 (7B)		2
Primary Cache Tag Array	PCTAG	124 (7C)	R/W	2
Primary Cache Index	PCIDX	125 (7D)	R/W	2
Primary Cache Error Address	PCERR	126 (7E)	R/W	2
Primary Cache Status	PCSTS	127 (7F)	R/W	2 I

Table 4–2 (Cont.): KA64A Internal Processor Registers

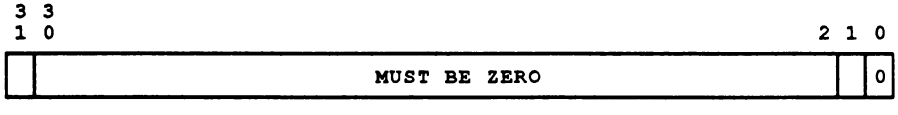
Register	Mnemonic	Address decimal (hex)	Type	Class
Reserved		128–143 (80–8F)		3
Reserved		144–147 (90–93)		2
Reserved		148–156 (94–9C)		5
Reserved		157–159 (9D–9F)		2
Reserved		160–255 (A0–FF)		3
Reserved		256 (100) and up		4

Figure 4–1: Machine Check Error Summary Register (MCESR) IPR38 (26 hex)



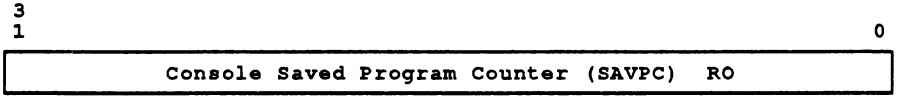
msb-p023-89

Figure 4–2: Accelerator Control and Status Register (ACCS) IPR40 (28 hex)



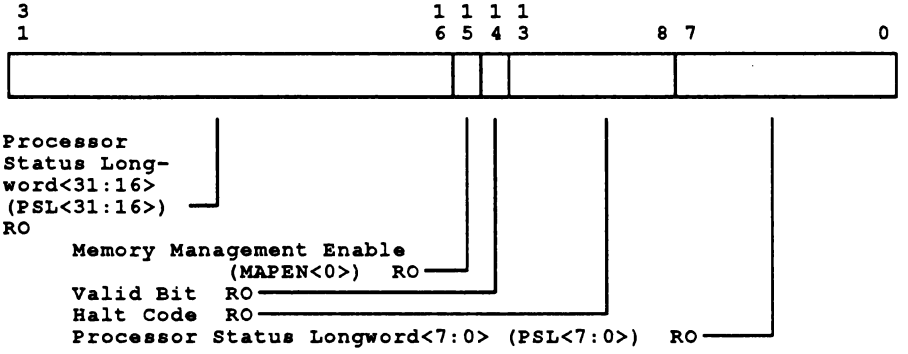
msb-p024-89

Figure 4–3: Console Saved Program Counter Register (SAVPC) IPR42 (2A hex)



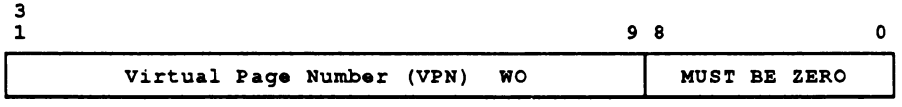
msb-p025-89

Figure 4–4: Console Saved Processor Status Longword (SAVPSL) IPR43 (2B hex)



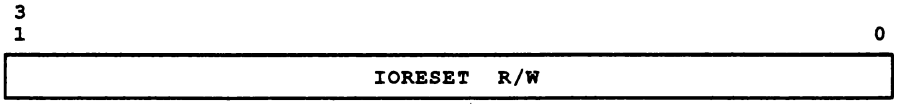
msb-p026-89

Figure 4–5: Translation Buffer Tag Register (TBTAG) IPR47 (2F hex)



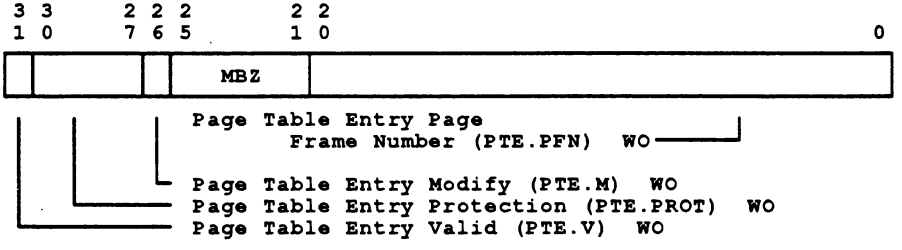
msb-p027-89

Figure 4–6: I/O Reset Register (IORESET) IPR55 (37 hex)



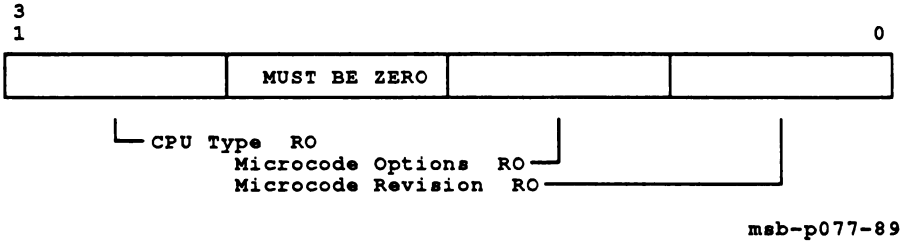
msb-p028-89

Figure 4–7: Translation Buffer Data Register (TBDATA) IPR59 (3B hex)



msb-p029-89

Figure 4–8: System Identification Register (SID) IPR62 (3E hex)



msb-p077-89

Figure 4–9: Backup Cache Backup Tag Store Register (BCBTS) IPR113 (71 hex)

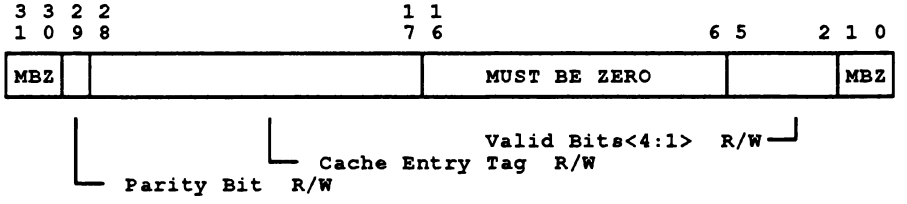


Figure 4–10: Backup Cache Primary 1 Tag Store Register (BCP1TS) IPR114 (72 hex)

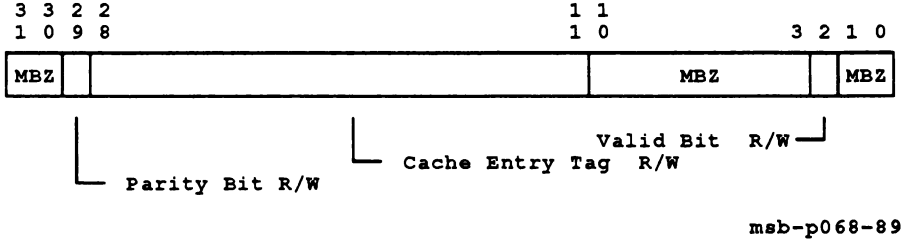


Figure 4–11: Backup Cache Primary 2 Tag Store Register (BCP2TS) IPR115 (73 hex)

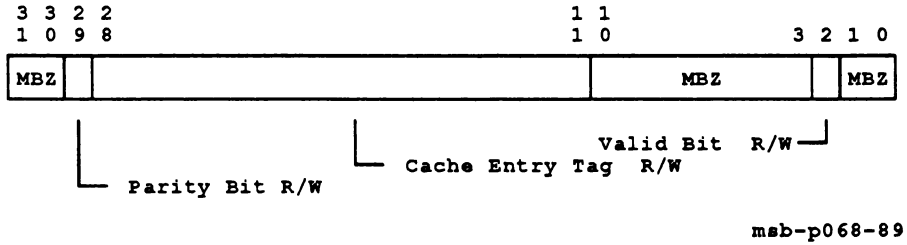
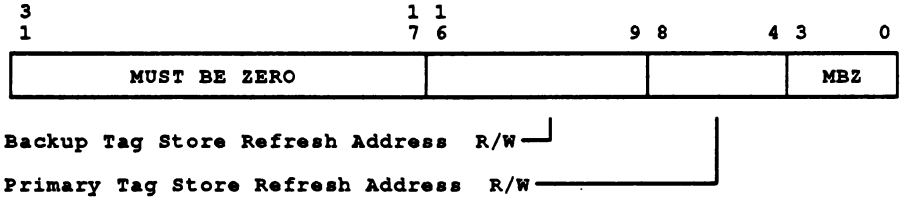


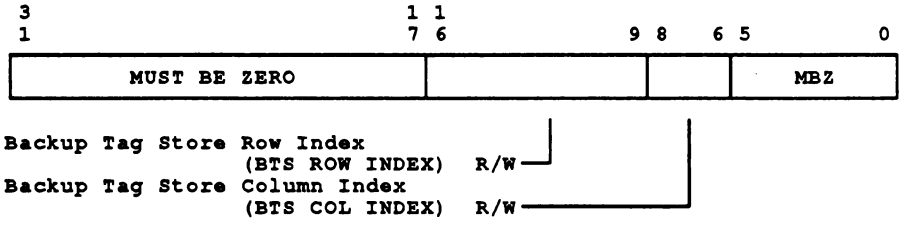
Figure 4–12: Backup Cache Refresh Register (BCRFR) IPR116 (74 hex)



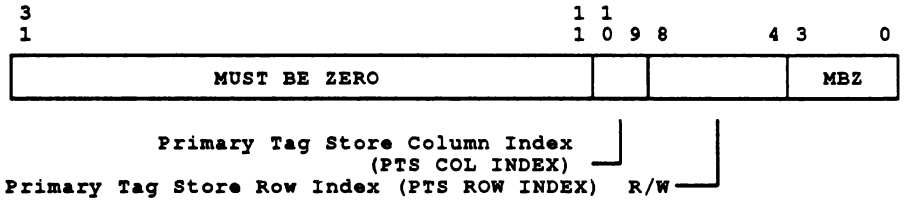
msb-p031-89

Figure 4–13: Backup Cache Index Register (BCIDX) IPR117 (75 hex)

When used for backup tag store indexing:

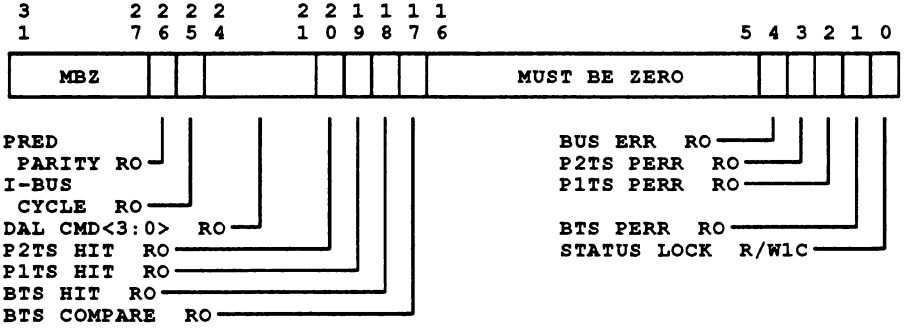


When used for primary tag store indexing:



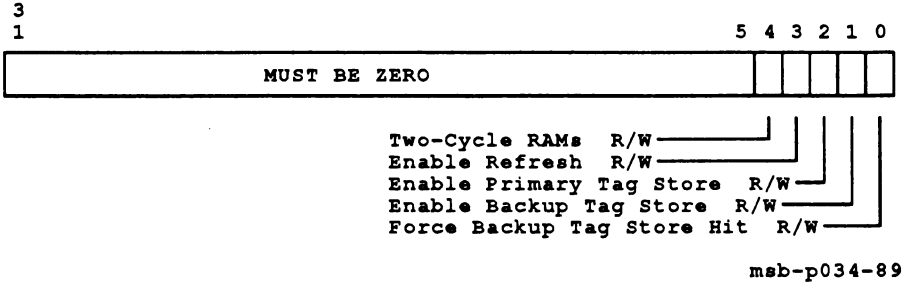
msb-p032-89

Figure 4–14: Backup Cache Status Register (BCSTS) IPR118 (76 hex)



msb-p033-89

Figure 4–15: Backup Cache Control Register (BCCTL) IPR119 (77 hex)



msb-p034-89

Figure 4–16: Backup Cache Error Address Register (BCERR) IPR120 (78 hex)

msb-p035-89

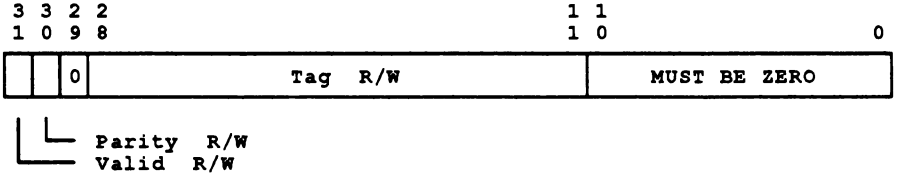
Figure 4–17: Backup Cache Flush Backup Tag Store Register (BCFBTS) IPR121 (79 hex)

msb-p036-89

Figure 4–18: Backup Cache Flush Primary Tag Store Register (BCFPTS) IPR122 (7A hex)

msb-p037-89

Figure 4–19: Primary Cache Tag Array Register (PCTAG) IPR124 (7C hex)



msb-p038-89

Figure 4–20: Primary Cache Index Register (PCIDX) IPR125 (7D hex)

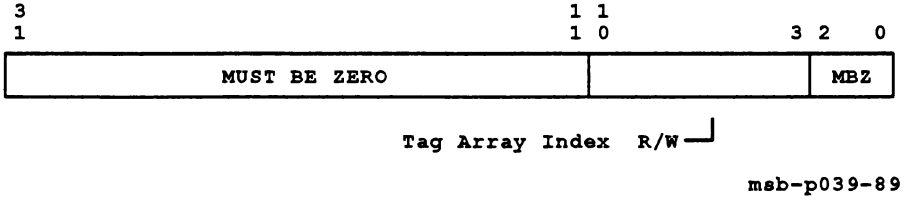
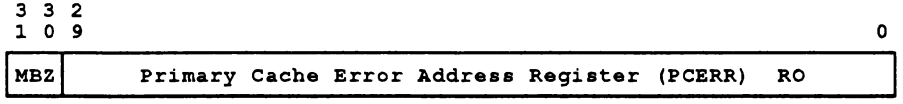


Figure 4–21: Primary Cache Error Address Register (PCERR) IPR126 (7E hex)

When Trap1 is set:



When Trap1 is not set:

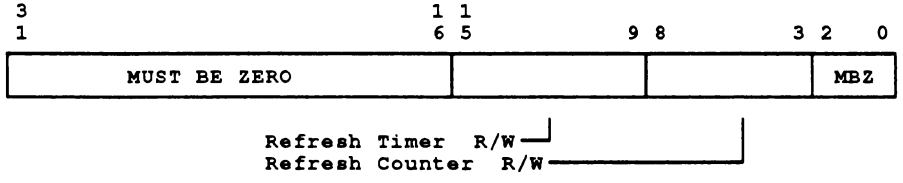
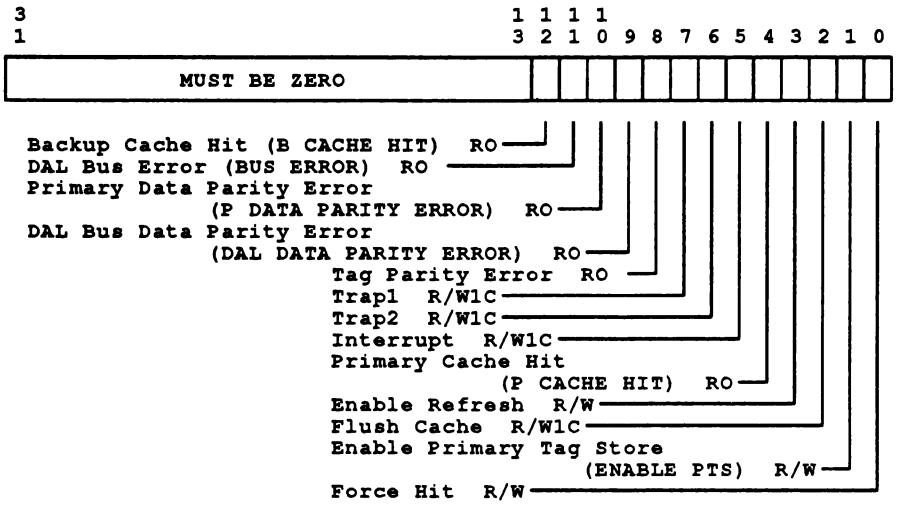


Figure 4–22: Primary Cache Status Register (PCSTS) IPR127 (7F hex)



msb-p041-89

4.2 KA64A Registers in XMI Private Space

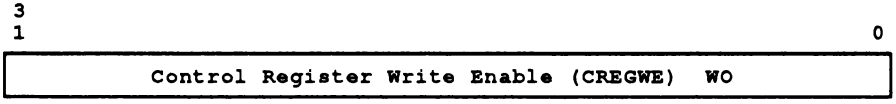
Table 4–3: KA64A Registers in XMI Private Space

Register	Mnemonic	Address
Control Register Write Enable	CREGWE	2000 0000
Console ROM (halt protected)		2004 0000 to 2007 FFFF
Console EEPROM (halt protected)		2008 0000 to 2008 7FFF
Console ROM (not halt protected)		200C 0000 to 200F FFFF
Console EEPROM (not halt protected)		2010 0000 to 2010 7FFF
RSSC Base Address	SSCBAR	2014 0000
RSSC Configuration	SSCCNR	2014 0010
RSSC Bus Timeout Control	SSCBTR	2014 0020
RSSC Output Port	OPORT	2014 0030
Control Register 0	CREG0	–
Control Register 1	CREG1	–
RSSC Input Port	IPORT	2014 0040
Control Register Base Address	CRBADR	2014 0130
Control Register Address Decode Mask	CRADMR	2014 0134
EEPROM Base Address	EEBADR	2014 0140
EEPROM Address Decode Mask	EEADMR	2014 0144
Timer 0 Control	TCR0	2014 0160
Timer 0 Interval	TIR0	2014 0164
Timer 0 Next Interval	TNIR0	2014 0168
Timer 0 Interrupt Vector	TIVR0	2014 016C
Timer 1 Control	TCR1	2014 0170
Timer 1 Interval	TIR1	2014 0174
Timer 1 Next Interval	TNIR1	2014 0178
Timer 1 Interrupt Vector	TIVR1	2014 017C
RSSC Interval Counter	SSCICR	2014 01F8

Table 4–3 (Cont.): KA64A Registers in XMI Private Space

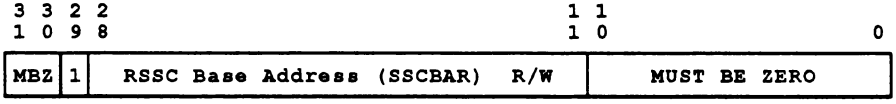
Register	Mnemonic	Address
RSSC Internal RAM		2014 0400 to 2014 07FF
IP IVINTR Generation	IPINTR	2101 0000 to 2101 FFFF
WE IVINTR Generation	WEINTR	2102 0000 to 2102 FFFF

Figure 4–23: Control Register Write Enable Register (CREGWE) 2000 0000



msb-p008-89

Figure 4–24: RSSC Base Address Register (SSCBAR) 2014 0000



msb-p009-89

Figure 4–25: RSSC Configuration Register (SSCCNR) 2014 0010

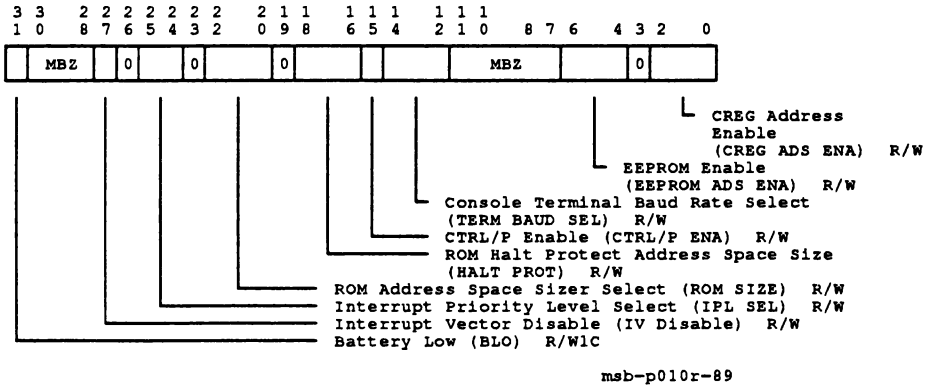


Figure 4–26: RSSC Bus Timeout Control Register (SSCBTR) 2014 0020

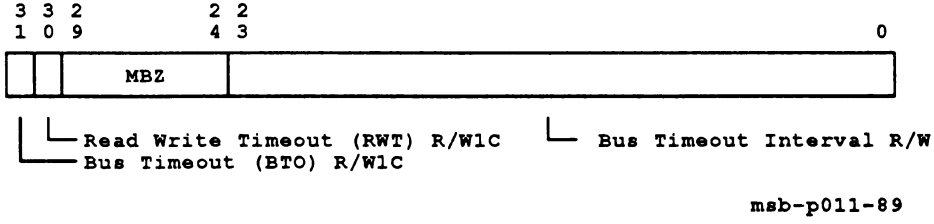


Figure 4–27: RSSC Output Port Register (OPORT) 2014 0030

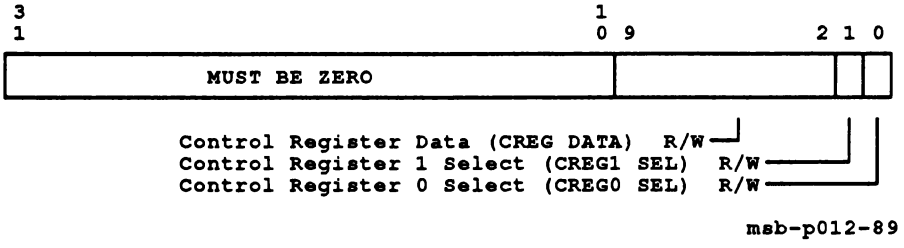


Figure 4–28: Control Register 0 (CREG0)

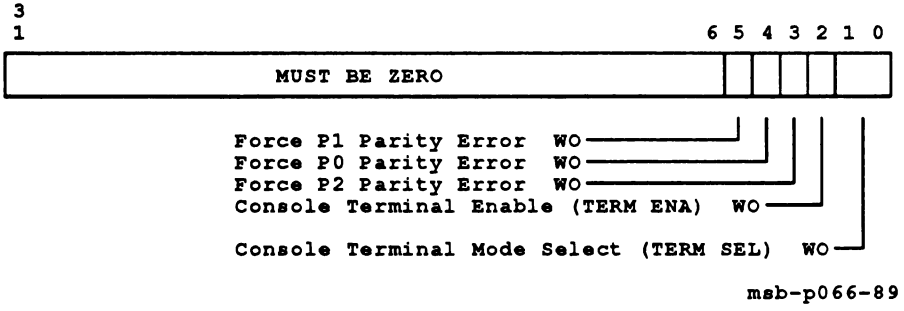


Figure 4–29: Control Register 1 (CREG1)

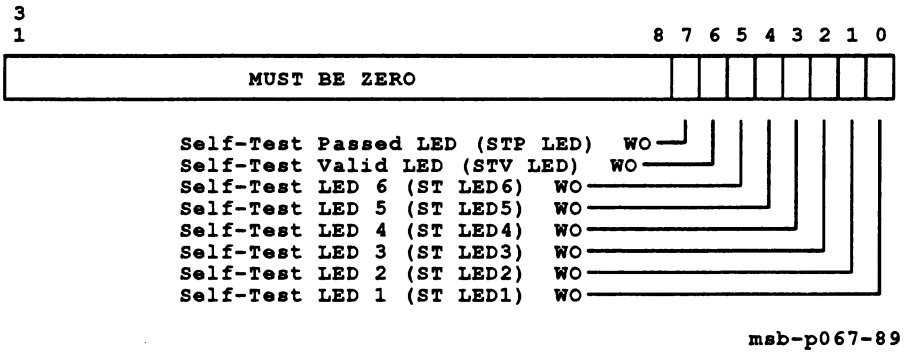


Figure 4–30: RSSC Input Port Register (IPORT) 2014 0040

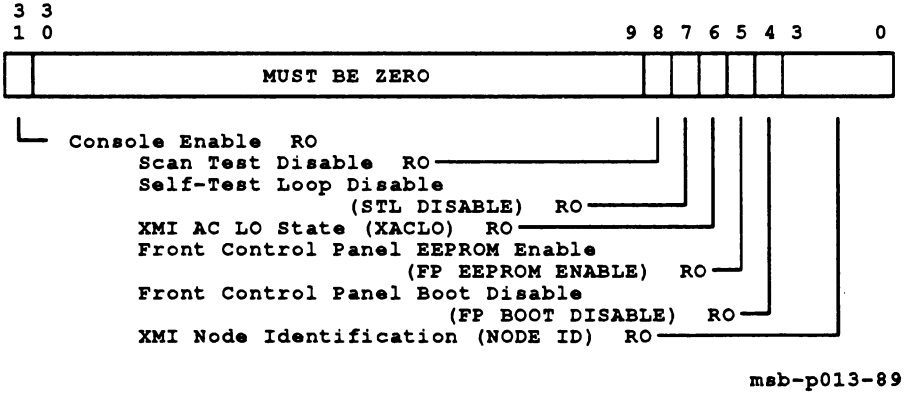


Figure 4–31: Control Register Base Address Register (CRBADR) 2014 0130

msb-p014-89

Figure 4–32: Control Register Address Decode Mask Register (CRADMR) 2014 0134

msb-p015-89

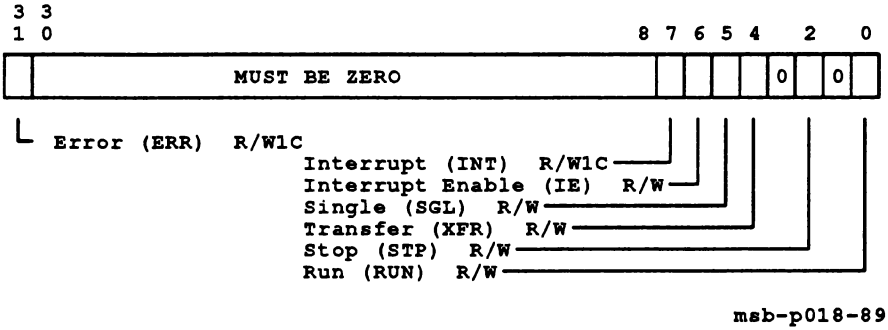
Figure 4–33: EEPROM Base Address Register (EEBADR) 2014 0140

msb-p016-89

Figure 4–34: EEPROM Address Decode Mask Register (EEADMR) 2014 0144

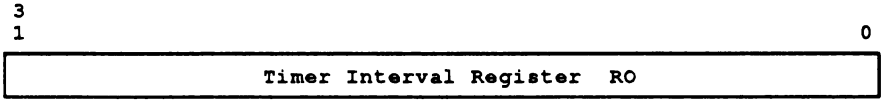
msb-p017-89

Figure 4–35: Timer Control Register 0 (TCR0)
2014 0160



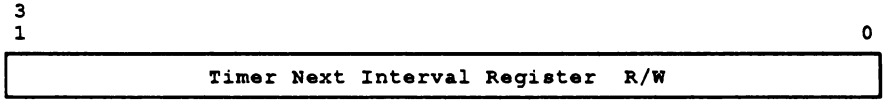
msb-p018-89

Figure 4–36: Timer Interval Register 0 (TIR0)
2014 0164



msb-p019-89

Figure 4–37: Timer Next Interval Register 0 (TNIR0)
2014 0168



msb-p020-89

Figure 4–38: Timer Interrupt Vector Register 0 (TIVR0) 2014 016C

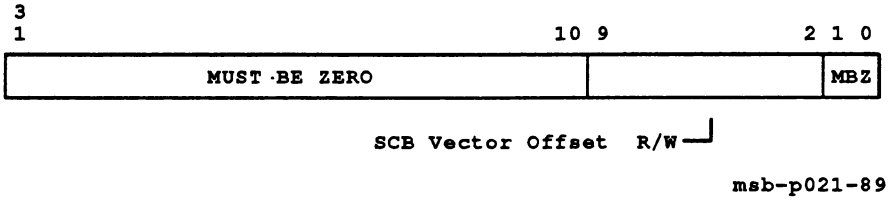


Figure 4–39: Timer Control Register 1 (TCR1) 2014 0170

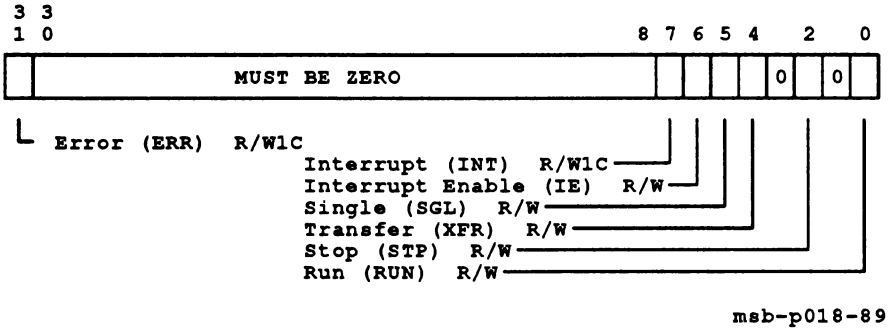


Figure 4–40: Timer Interval Register 1 (TIR1) 2014 0174

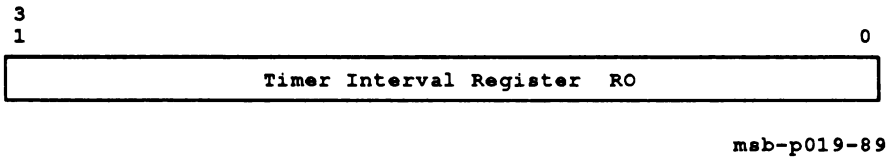
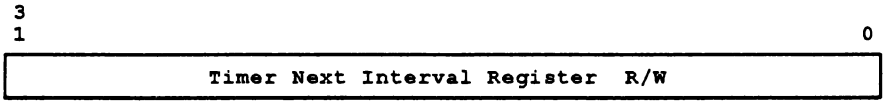
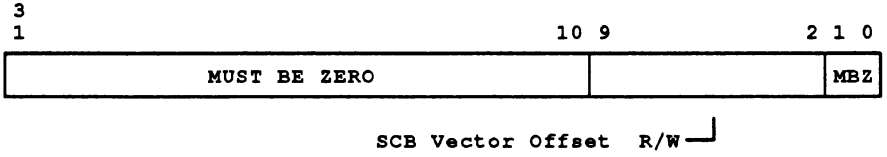


Figure 4–41: Timer Next Interval Register 1 (TNIR1)
2014 0178



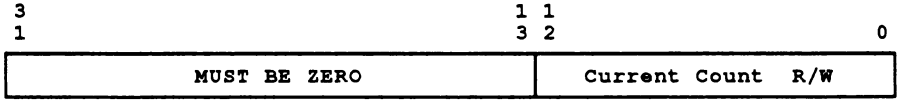
msb-p020-89

Figure 4–42: Timer Interrupt Vector Register 1 (TIVR1)
2014 017C



msb-p021-89

Figure 4–43: Interval Counter Register (SSCICR)
2014 01F8



msb-p022-89

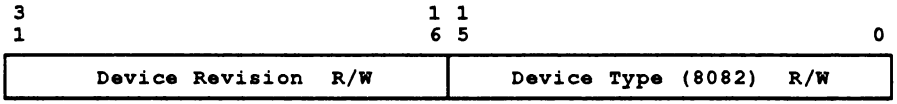
4.3 KA64A XMI Registers

Table 4–4: XMI Registers for the KA64A CPU Module

Name	Mnemonic	Address
XMI Device Register	XDEV	BB[1]+ 00
XMI Bus Error Register	XBER	BB + 04
XMI Failing Address Register	XFADR	BB + 08
XMI General Purpose Register	XGPR	BB + 0C
KA64A Control and Status Register	RCSR	BB + 10

[1]BB = base address of an XMI node, which is the address of the first location in nodespace.

Figure 4–44: Device Register (XDEV) BB + 00



msb-p003-89

Figure 4–45: Bus Error Register (XBER)
BB + 04

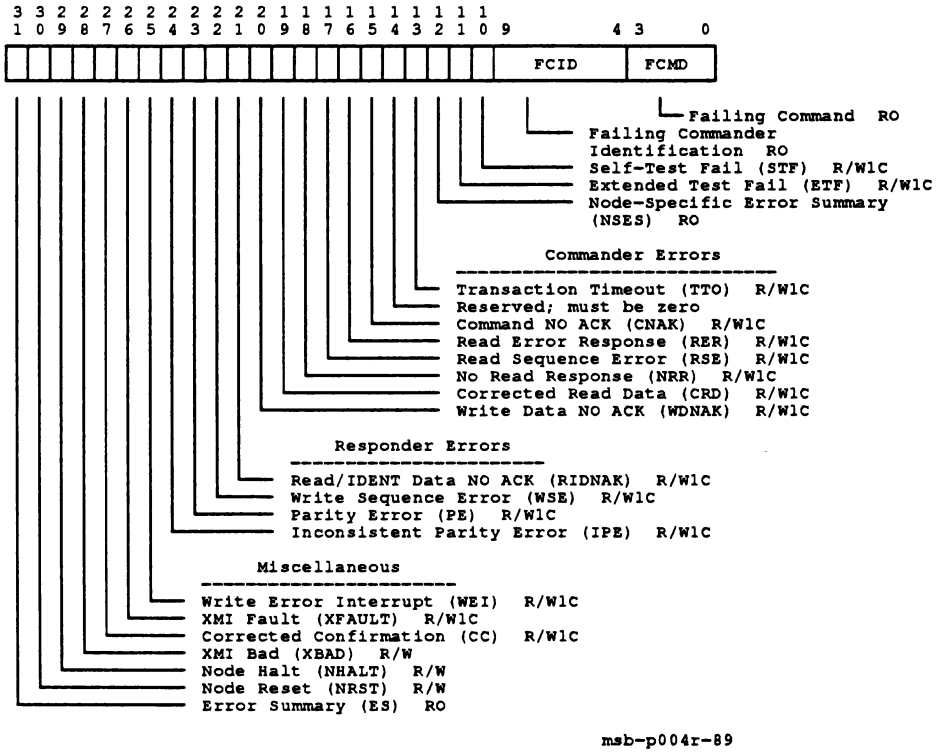
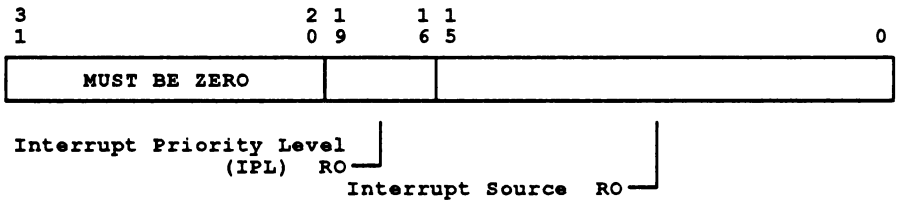
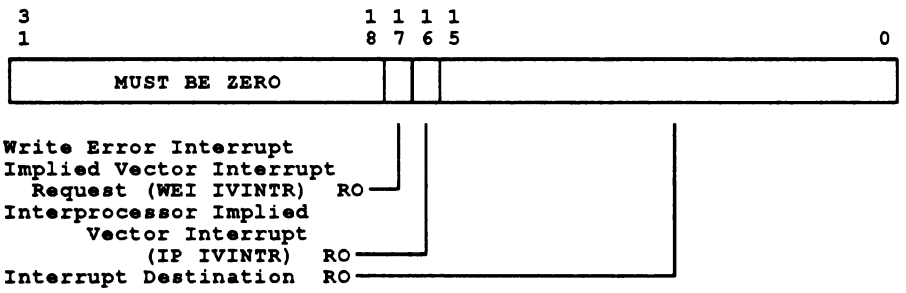


Figure 4–46: Failing Address Register (XFADR)
BB + 08

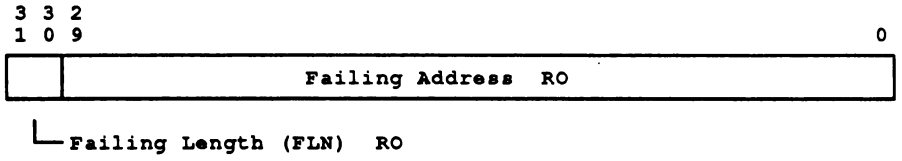
When XBER <3:0> = 1001 (IDENT transaction):



When XBER <3:0> = 1111 (IVINTR transaction):



When XBER <3:0> is neither an IDENT transaction nor an IVINTR transaction:

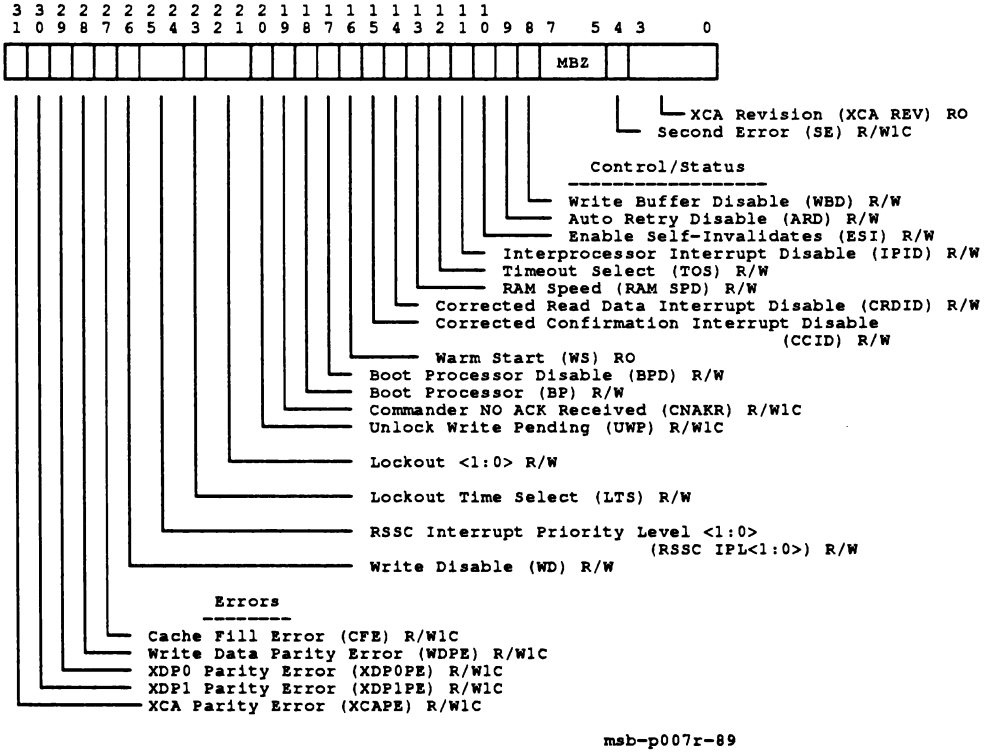


msb-p005-89

Figure 4–47: XMI General Purpose Register (XGPR) BB + 0C

msb-p006-89

Figure 4–48: KA64A Control and Status Register (RCSR) BB + 10



msb-p007r-89

4.4 Machine Checks

Figure 4–49 shows the parameters that are pushed on the stack in response to a machine check. Table 4–5 lists these parameters. Note that machine checks are taken regardless of the current IPL. If the machine check exception vector bits (<1:0>) are not both one, the operation of the processor is undefined. The exception is taken on the interrupt stack and the IPL is raised to 1F (hex).

Figure 4–49: The Stack in Response to a Machine Check

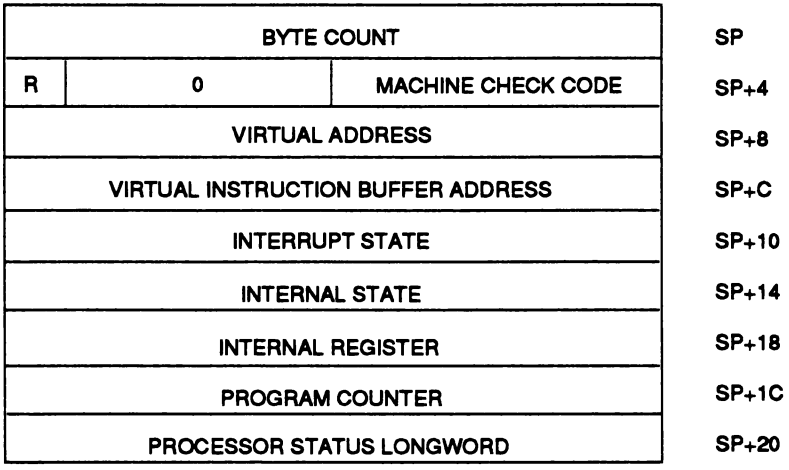


Table 4–5: Machine Check Parameters

Parameter	Value (hex) or Bit	Description
Byte Count (SP)	18	Size of stack frame in bytes, not including PSL, PC, or byte count longword
Machine check code (SP+4)	01	Floating-point operand or result transfer error
	02	Floating-point reserved instruction
	03	Floating-point operand parity error
	04	Floating-point unknown status error
	05	Floating-point returned result parity error
	08	Translation buffer miss in ACV/TNV microflow
	09	Translation buffer hit in ACV/TNV microflow
	0A	Undefined INT.ID value
	0B	Undefined MOVCx state
	0C	Undefined instruction trap code
	0D	Undefined control store address
	10	Cache read tag/data parity error
	11	DAL bus or data parity read error
	12	DAL bus error on write or clear write buffer
	13	Undefined bus error microtrap
Virtual address (SP+8)	<31:0>	Current contents of VAP register
Virtual instruction buffer address (SP+C)	<31:0>	Current virtual instruction buffer address
Interrupt state (SP+10)	<22>	ICCS bit <6>
	<15:1>	SISR bits <15:1>
Internal state (SP+14)	<31:24>	Difference between current PC and PC of opcode
	<20:18>	Address of last memory reference

Table 4–5 (Cont.): Machine Check Parameters

Parameter	Value (hex) or Bit	Description
	<17:16>	Data length of last memory reference
	<15:8>	Opcode
	<3:0>	Reserved
Internal register (SP+18)	<31:0>	
Program counter (SP+1C)	<31:0>	PC at the start of the current instruction
Processor status longword (SP+20)	<31:0>	Current contents of PSL

4.5 Parse Trees

Figure 4–50: Machine Check Parse Tree

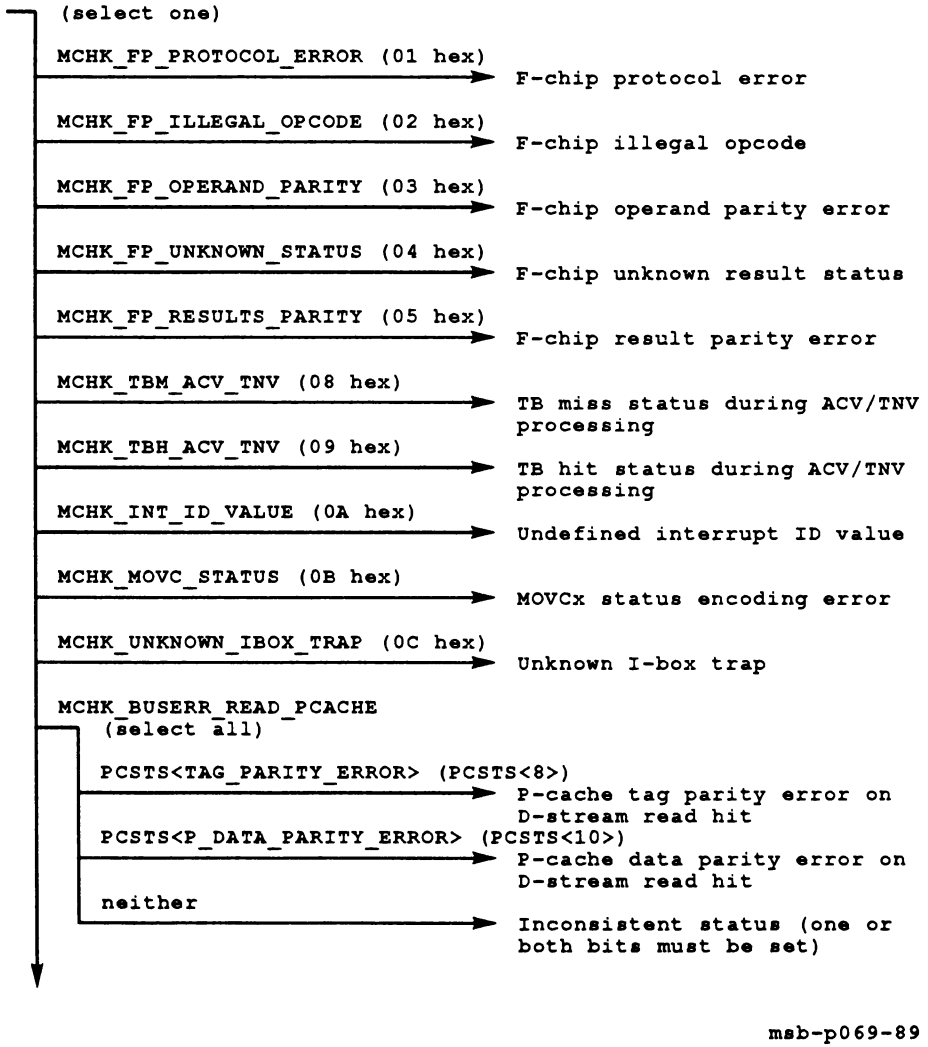


Figure 4–50 Cont'd. on next page

Figure 4–50 (Cont.): Machine Check Parse Tree

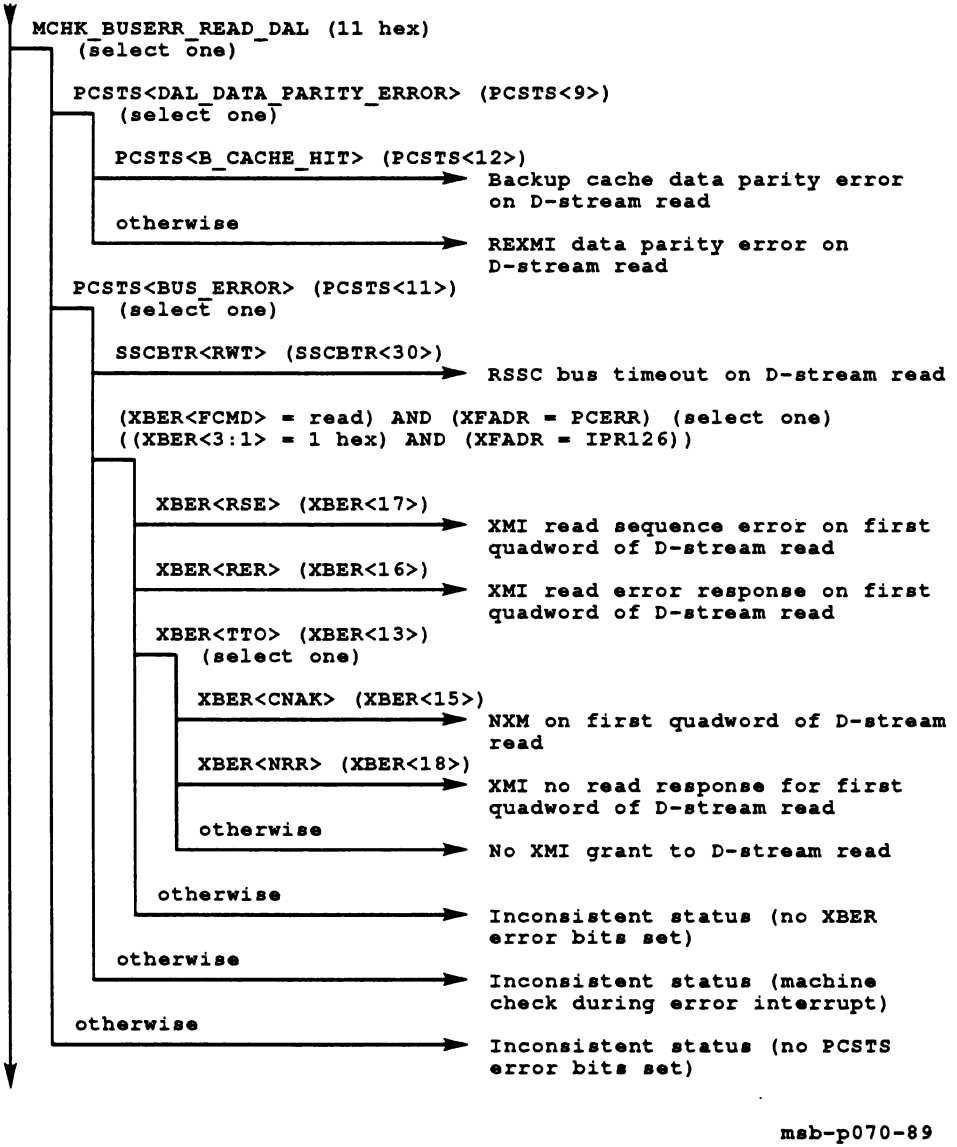
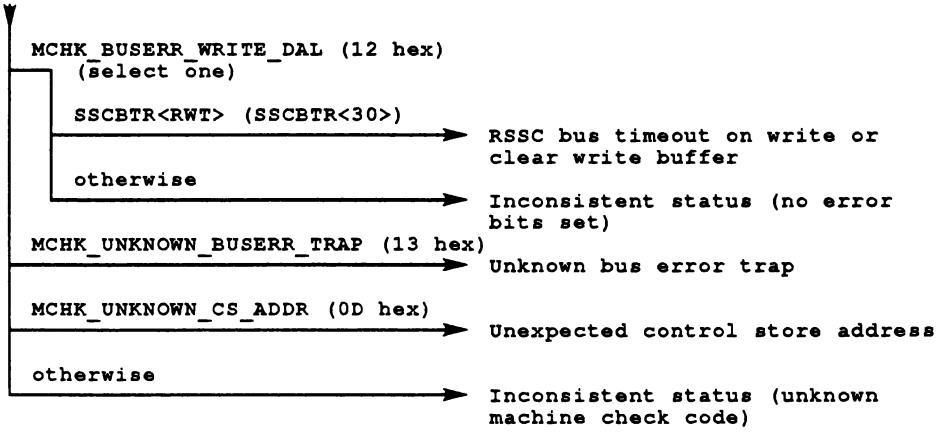


Figure 4–50 Cont'd. on next page

Figure 4–50 (Cont.): Machine Check Parse Tree



NOTES:

(select one) - exactly one case must be true. If zero or more than one is true, the status is inconsistent.

(select all) - more than one case may be true.

otherwise - fall-through case for (select one) if no other options are true.

neither - fall-through case for (select all) if none of the options are true.

The parse tree assumes that retry is enabled (RCSR<ARD> = 0).

msb-p071-89

Figure 4–51: Hard Error Interrupt Parse Tree

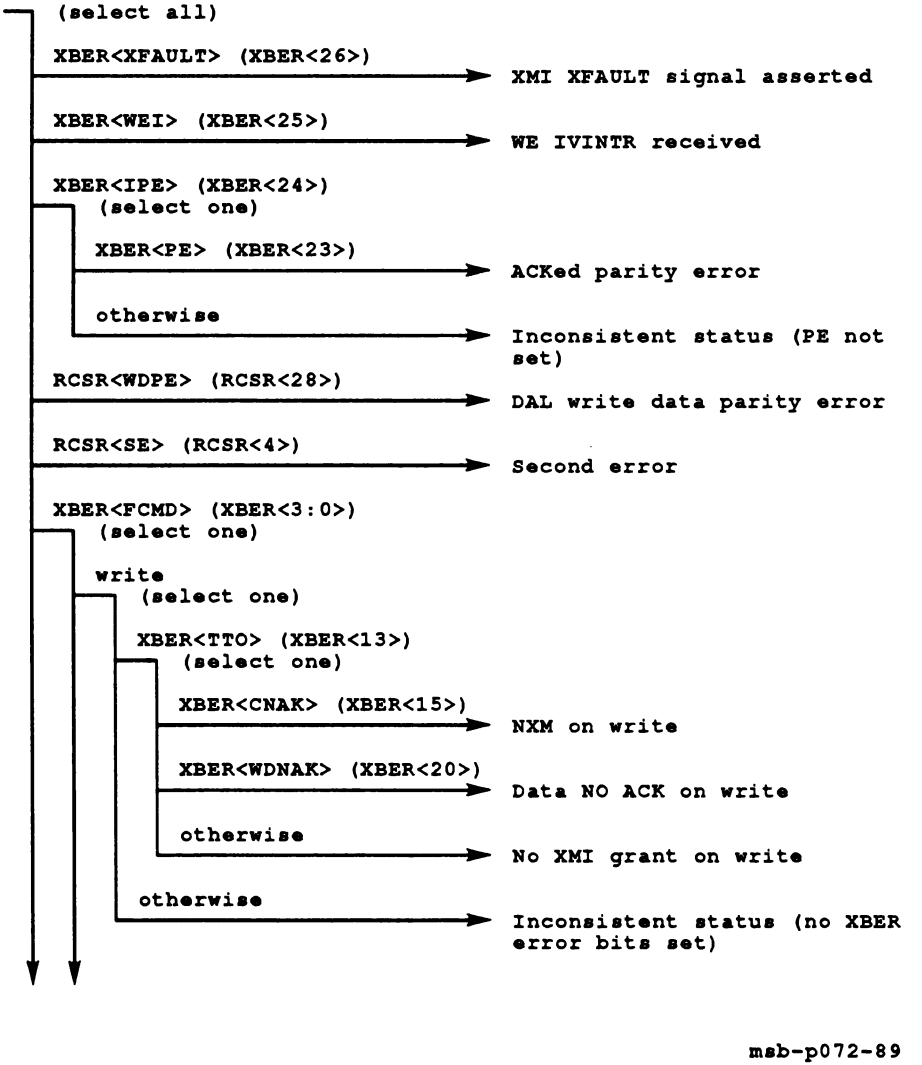
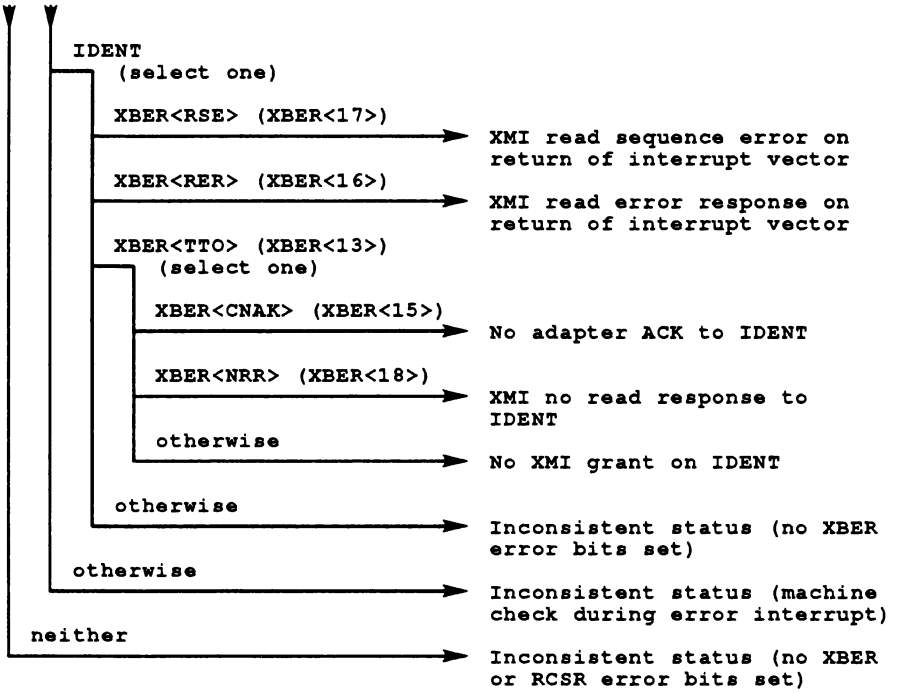


Figure 4–51 Cont'd. on next page

Figure 4–51 (Cont.): Hard Error Interrupt Parse Tree



NOTES:

(select one) - only one case must be true. If none or more than one is true, the status is inconsistent.

(select all) - more than one case may be true.

neither - fall-through case for (select all) if none of the options are true.

otherwise - fall-through case for (select one) if no other options are true.

The parse tree assumes that retry is enabled (RCSR<ARD> = 0).

msb-p073-89

Figure 4–52: Soft Error Interrupt Parse Tree

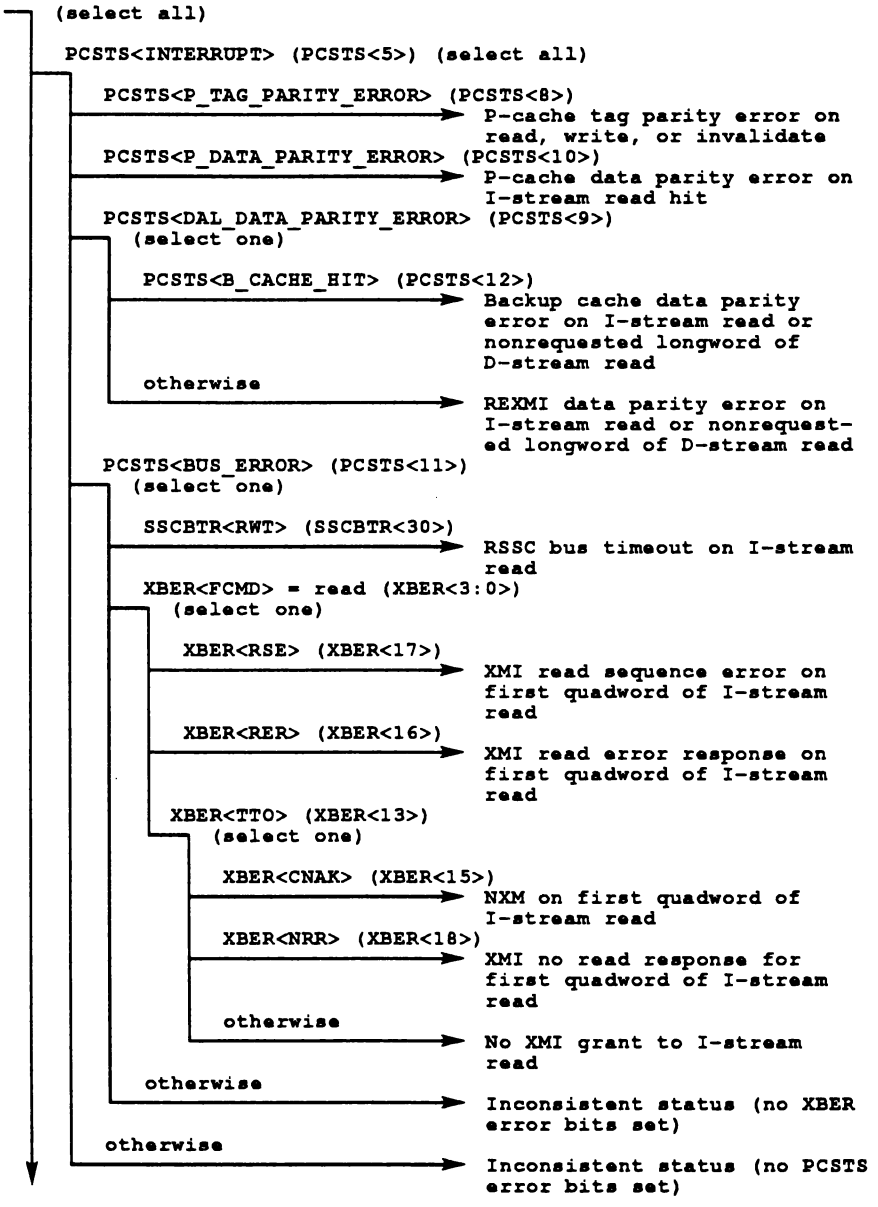
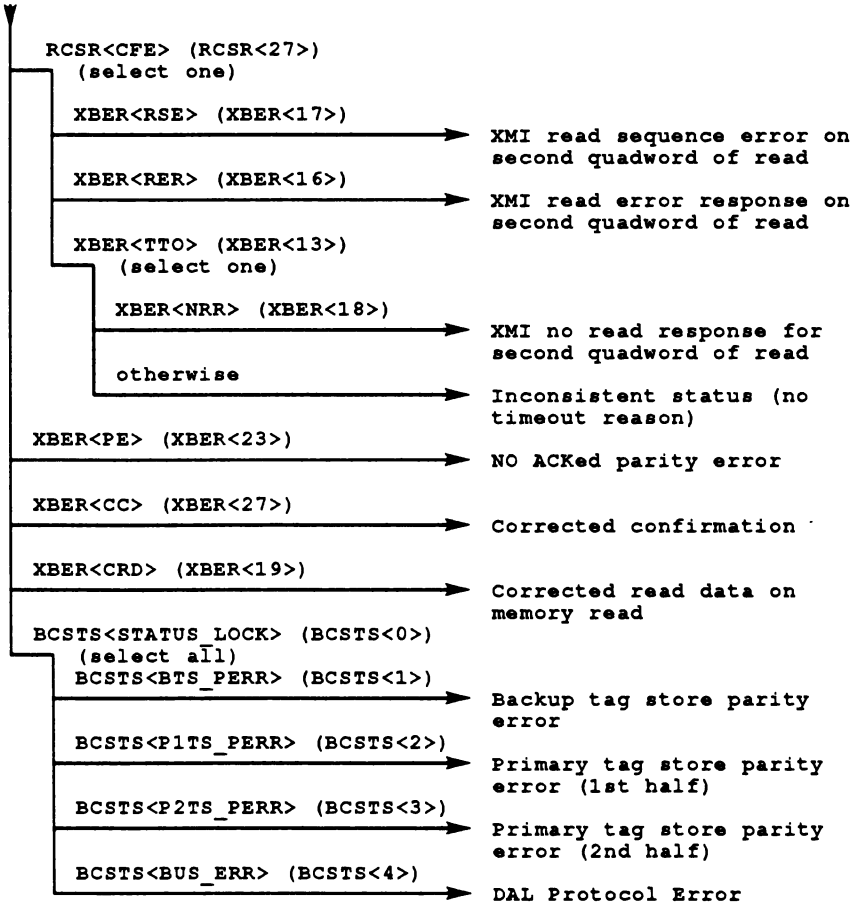


Figure 4–52 Cont'd. on next page

msb-p074r-89

Figure 4–52 (Cont.): Soft Error Interrupt Parse Tree



NOTES:

(select one) - exactly one case must be true. If none or more than one is true, the status is inconsistent.

(select all) - more than one case may be true.

otherwise - fall-through case for (select one) if no other options are true.

The parse tree assumes that retry is enabled (RCSR<ARD> = 0).

msb-p075-89

Chapter 5
MS62A Memory Registers

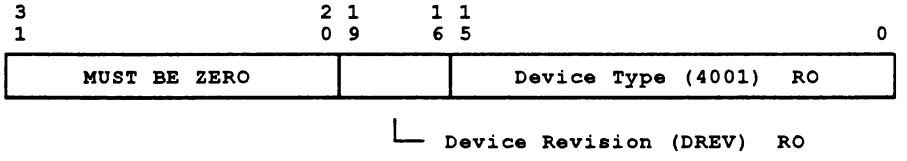
Table 5–1: MS62A Control and Status Registers

Name	Mnemonic	Address
Device Register	XDEV	BB[1]+ 00
Bus Error Register	XBER	BB + 04
Starting and Ending Address Register	SEADR	BB + 10
Memory Control Register 1	MCTL1	BB + 14
Memory ECC Error Register	MECER	BB + 18
Memory ECC Error Address Register	MECEA	BB + 1C
Memory Control Register 2	MCTL2	BB + 30
TCY Tester Register	TCY	BB + 34
Interlock Flag Status Register	IFLGn	BB + n[2]

[1]"BB" refers to the base address of the XMI node, which is the address of the first location in nodespace (see Table 3–1.)

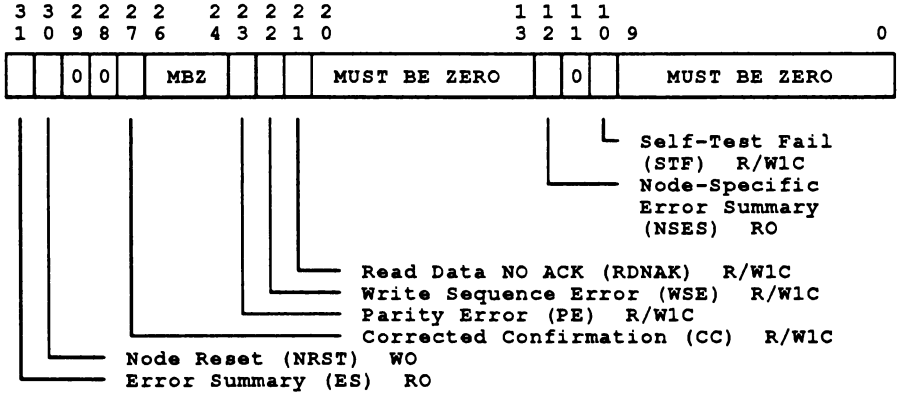
[2]"n" depends on the Interlock Flag Status Register number (0–15)

Figure 5–1: Device Register (XDEV)
BB + 00



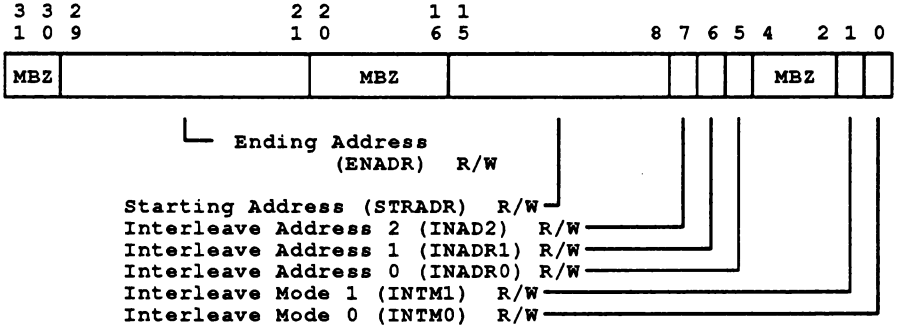
msb-p042-89

Figure 5–2: Bus Error Register (XBER)
BB + 04



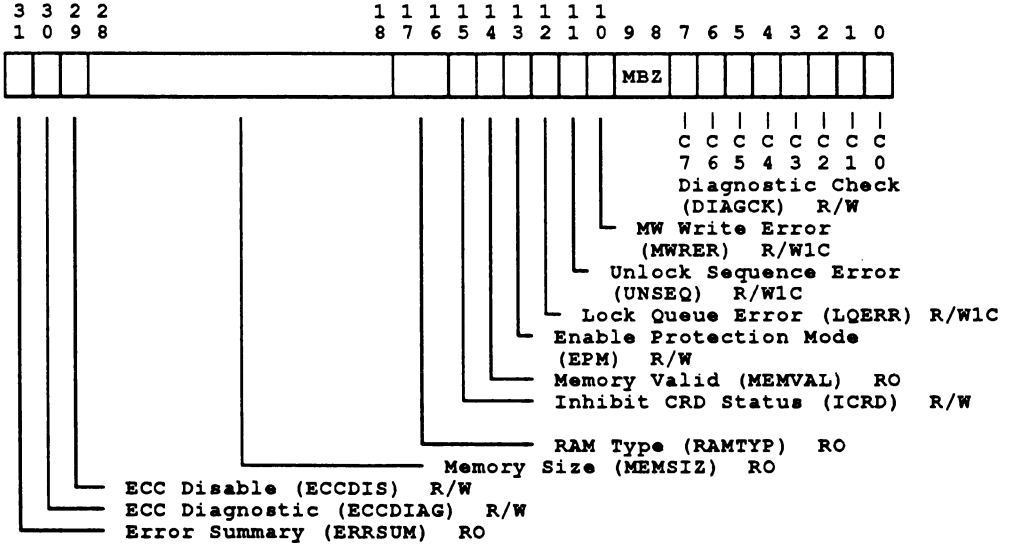
msb-p043-89

Figure 5–3: Starting and Ending Address Register (SEADR)
BB + 10



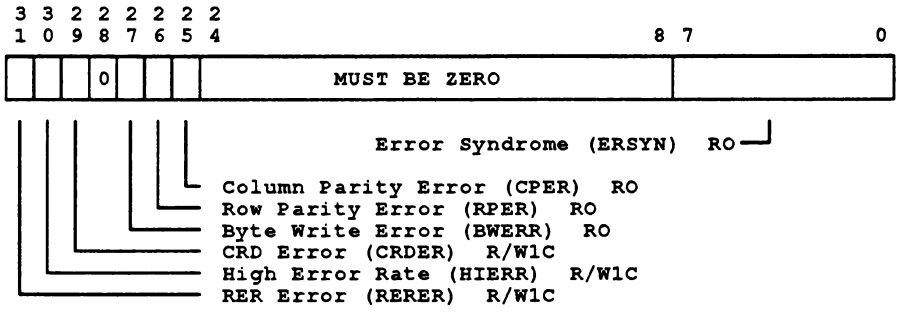
msb-p044-89

Figure 5–4: Memory Control Register 1 (MCTL1)
BB + 14



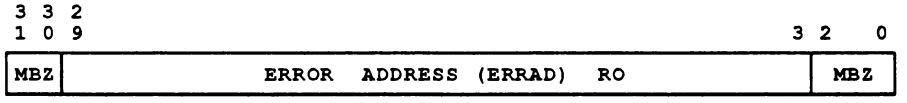
msb-p045-89

Figure 5–5: Memory ECC Error Register (MECER)
BB + 18



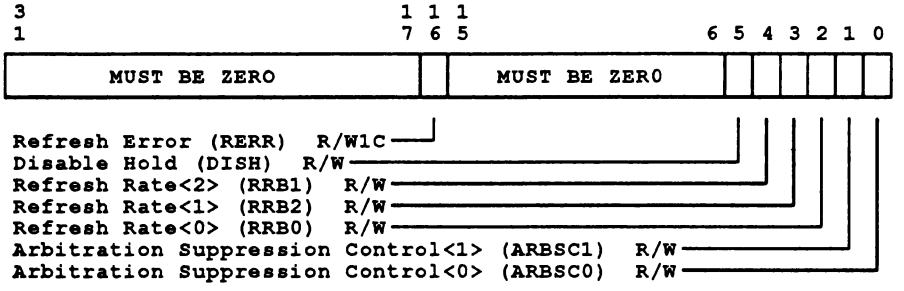
msb-p046-89

Figure 5–6: Memory ECC Error Address Register (MECEA)
BB + 1C



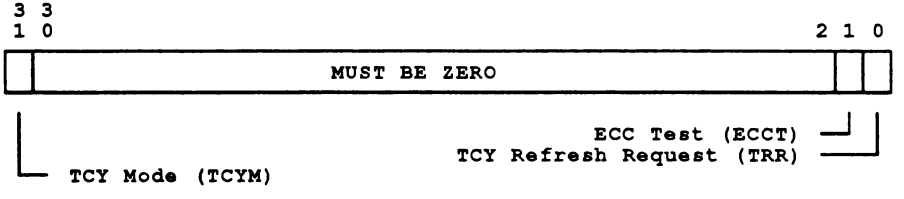
msb-p047-89

Figure 5–7: Memory Control Register 2 (MCTL2)
BB + 30



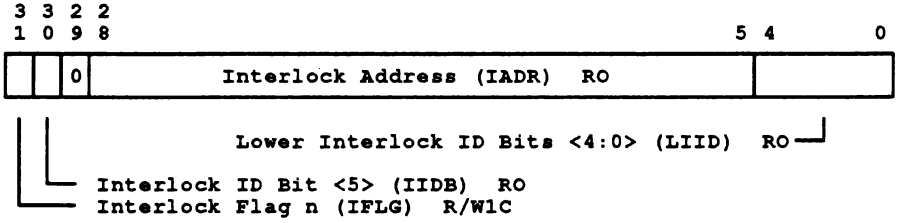
msb-p048-89

Figure 5–8: TCY Tester Register (TCY)
BB + 34



msb-p049-89

Figure 5–9: Interlock Flag Status Registers (IFLGn)
BB + relative address



where n (0-15) is the number of the Interlock Flag Register:

Register	Relative Address
IFLG0	BB + 20
IFLG1	BB + 24
IFLG2	BB + 28
IFLG3	BB + 2C
IFLG4	BB + 40
IFLG5	BB + 44
IFLG6	BB + 48
IFLG7	BB + 4C
IFLG8	BB + 80
IFLG9	BB + 84
IFLG10	BB + 88
IFLG11	BB + 8C
IFLG12	BB + 100
IFLG13	BB + 104
IFLG14	BB + 108
IFLG15	BB + 10C

msb-p050-89

Chapter 6
DWMBA Adapter Registers

The DWMBA adapter consists of two modules: an XMI module in the XMI card cage and a VAXBI module in the VAXBI card cage. Table 6–1 lists the DWMBA registers: some of which are XMI required registers, some DWMBA/A registers, some DWMBA/B registers, and the VAXBI Device Register for the DWMBA/B module.

Register addresses for a particular device in a system are found by adding an offset to the base address for that device. To distinguish between addresses in VAXBI address space and addresses in XMI address space, we use the following convention:

lowercase bb + offset indicates an address in VAXBI address space
uppercase BB + offset indicates an address in XMI address space

Table 6–1: DWMBA Registers

Name	Mnemonic[1]	Address[2]
Device Register	XDEV	BB + 00
Bus Error Register	XBER	BB + 04
Failing Address Register	XFADR	BB + 08
Responder Error Address Register	AREAR	BB + 0C
DWMBA/A Error Summary Register	AESR	BB + 10
Interrupt Mask Register	AIMR	BB + 14
Implied Vector Interrupt Destination/Diagnostic Register	AIVINTR	BB + 18
Diag 1 Register	ADG1	BB + 1C
Control and Status Register	BCSR	BB + 40
DWMBA/B Error Summary Register	BESR	BB + 44
Interrupt Destination Register	BIDR	BB + 48
Timeout Address Register	BTIM	BB + 4C
Vector Offset Register	BVOR	BB + 50
Vector Register	BVR	BB + 54
Diagnostic Control Register 1	BDCR1	BB + 58
Reserved Register	–	BB + 5C
Device Register[3]	DTYPE	bb + 00

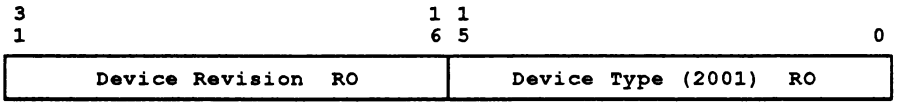
[1]The first letter of the mnemonic indicates the following:

X=XMI register, resides on the DWMBA/A module
A=Resides on the DWMBA/A module
B=Resides on the DWMBA/B module; accessible from the XMI bus

[2]The abbreviation "BB" refers to the base address of an XMI node (the address of the first location of the nodespace). The abbreviation "bb" refers to the base address in VAXBI nodespace.

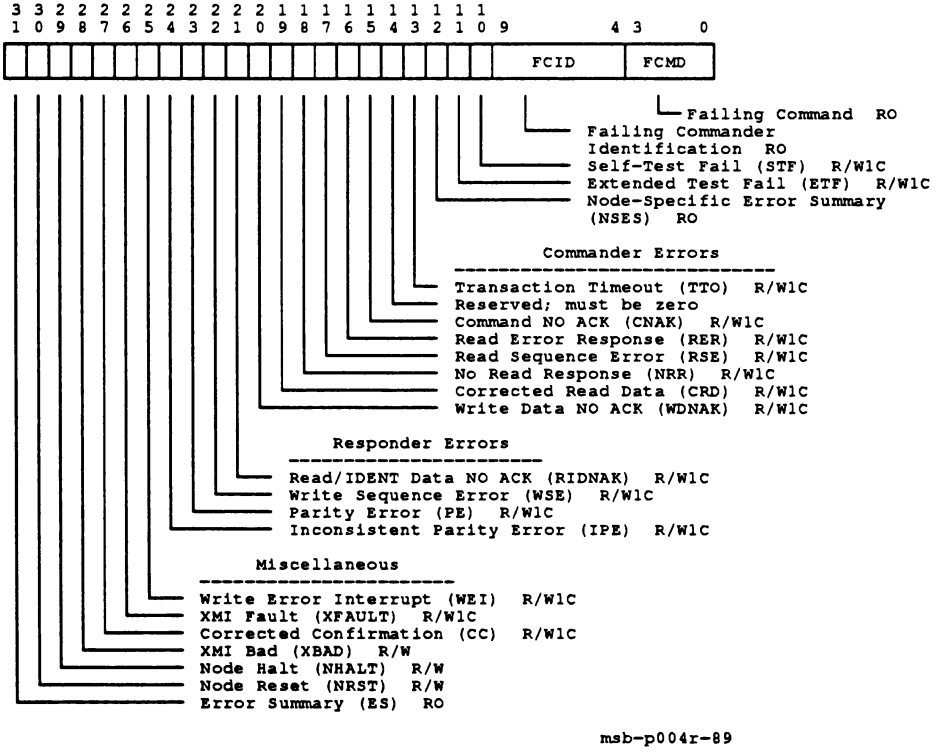
[3]This is a VAXBI register. For information on other VAXBI registers, see the *VAXBI Options Handbook*.

Figure 6–1: Device Register (XDEV)
BB + 00



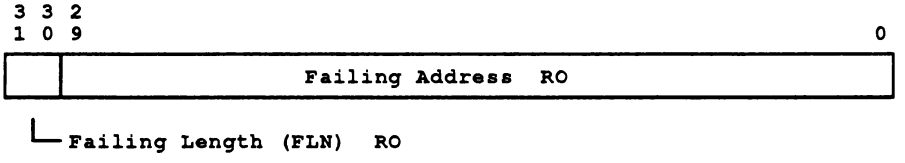
msb-p051-89

Figure 6–2: Bus Error Register (XBER)
BB + 04



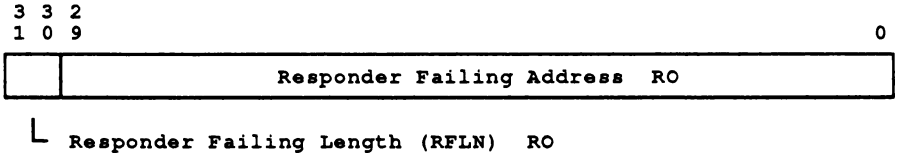
msb-p004r-89

Figure 6–3: Failing Address Register (XFADR)
BB + 08



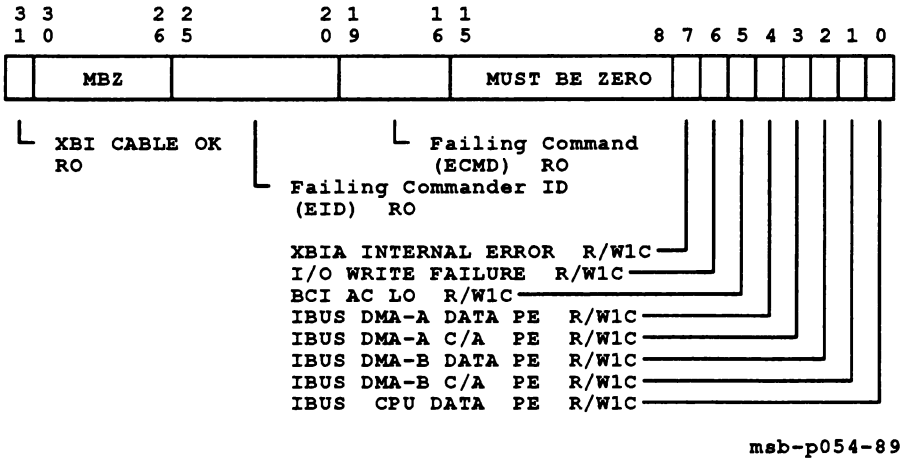
msb-p052-89

Figure 6–4: Responder Error Address Register (AREAR)
BB + 0C



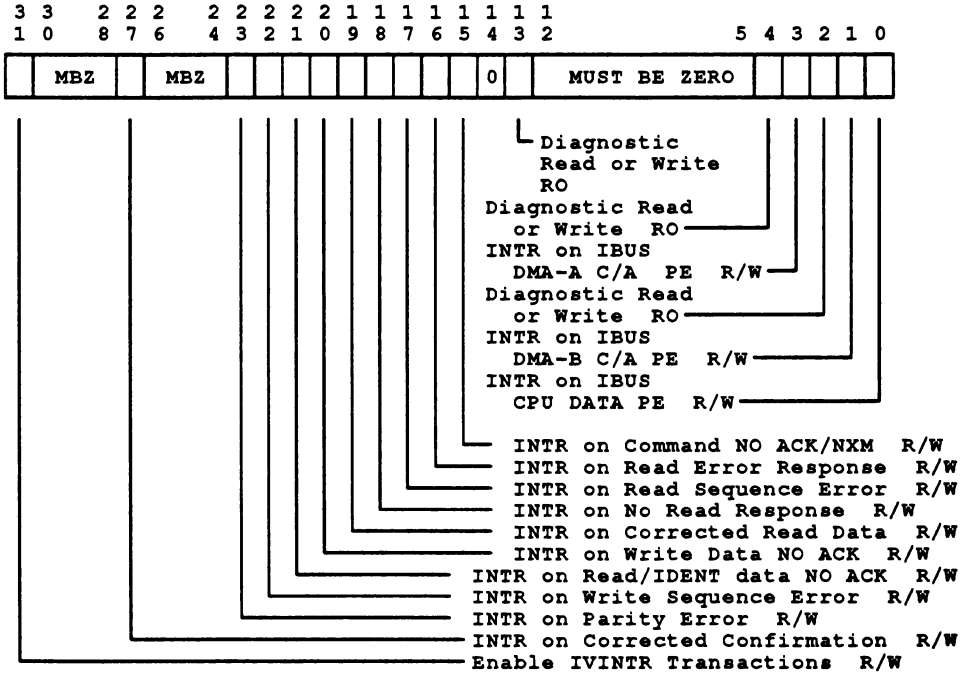
msb-p053-89

Figure 6–5: DWMBA/A Error Summary Register (AESR)
BB + 10



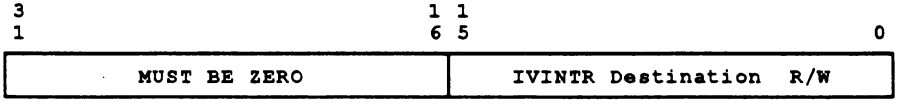
msb-p054-89

Figure 6–6: Interrupt Mask Register (AIMR) BB + 14



msb-p055-89

Figure 6–7: Implied Vector Interrupt Destination/Diagnostic Register (AIVINTR) BB + 18



msb-p056-89

Figure 6–8: Diag 1 Register (ADG1)
BB + 1C

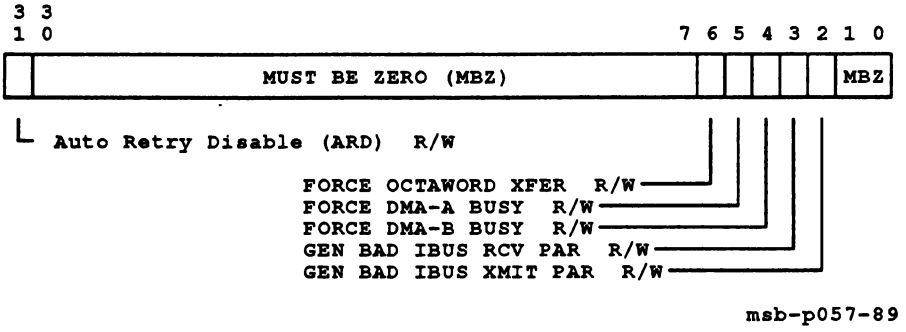


Figure 6–9: Control and Status Register (BCSR)
BB + 40

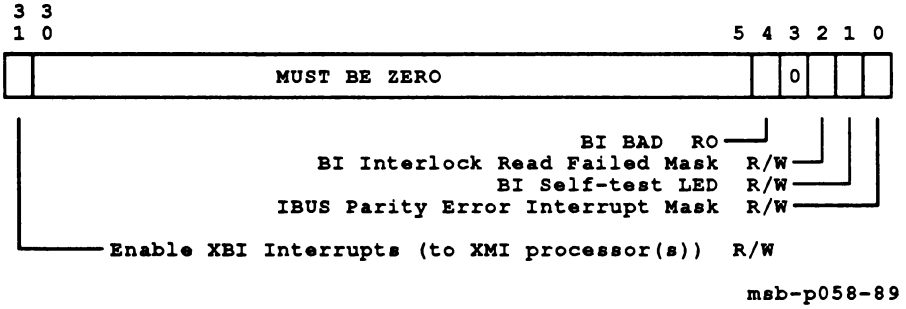
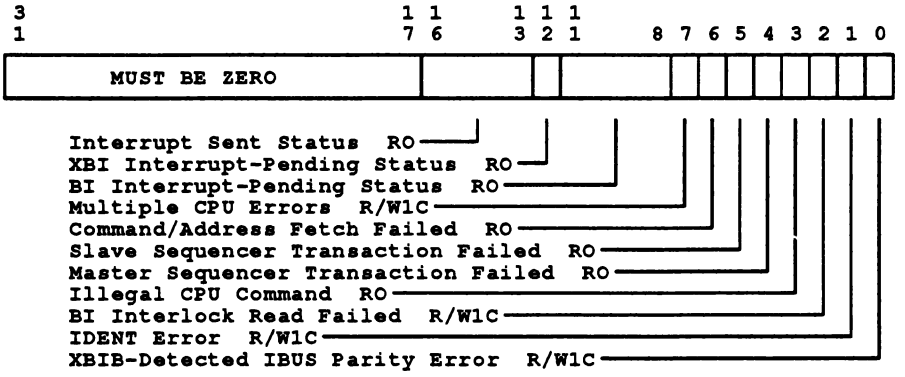
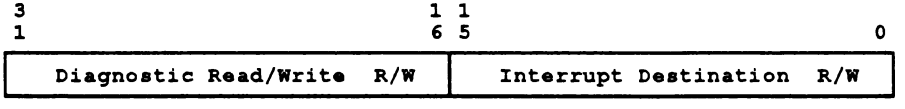


Figure 6–10: DWMBA/B Error Summary Register (BESR)
BB + 44



msb-p059-89

Figure 6–11: Interrupt Destination Register (BIDR)
BB + 48

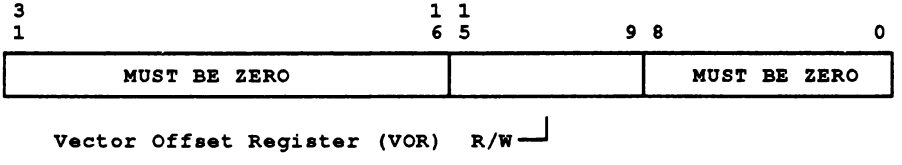


msb-p060-89

Figure 6–12: Timeout Address Register (BTIM)
BB + 4C

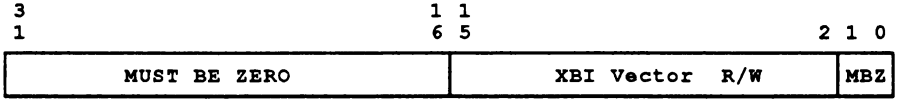
msb-p061-89

Figure 6–13: Vector Offset Register (BVOR)
BB + 50



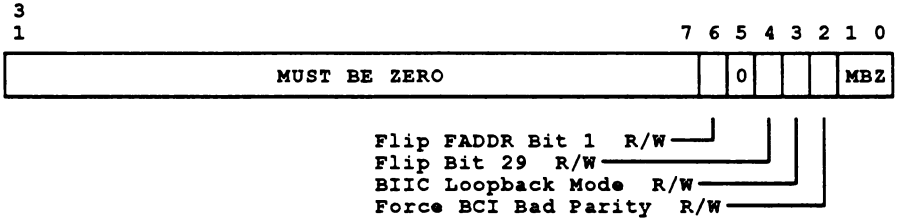
msb-p062-89

Figure 6–14: Vector Register (BVR)
BB + 54



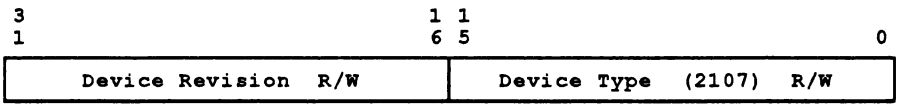
msb-p063-89

Figure 6–15: Diagnostic Control Register 1 (BDCR1)
BB + 58



msb-p064-89

Figure 6–16: VAXBI Device Register (DTYPE)
bb + 00



msb-p065-89

Index

A

B

C

D

E

F

H

I

K

M

O

P

R

S

T

V

X

digital