VAX 6000 Model 600 Mini-Reference

Order Number EK–660EA–HR.001

This manual supplies easy-to-access key information on VAX 6000 Model 600 systems.

digital equipment corporation
maynard, massachusetts

First Printing, January 1992

Contents

Index

Examples

Figures

Tables

Preface

Intended Audience

This manual is intended for the system manager, system programmer, and customer service engineers.

Document Structure

This manual has six chapters:

- **Chapter 1—Console Operation**
- **Chapter 2—Self-Test**
- **Chapter 3—Address Space**
- **Chapter 4—KA66A CPU Module Registers**
- **Chapter 5—MS66A Memory Registers**
- **Chapter 6—DWMBB Adapter Registers**

VAX 6000 Series Documents

There are two sets of documentation: manuals that apply to all VAX 6000 series systems and manuals that are specific to one VAX 6000 model. Table 1 lists the manuals in the VAX 6000 series documentation set.

Table 1: VAX 6000 Series Documentation

Title	Order Number
Operation	
VAX 6000 Series Owner's Manual	EK–600EB–OM
Service and Installation	
VAX 6000 Platform Technical User's Guide	EK–600EA–TM
VAX 6000 Series Installation Guide	EK–600EB–IN
VAX 6000 Installationsanleitung	EK–600GB–IN
VAX 6000 Guide d'installation	EK–600FB–IN
VAX 6000 Guia de instalacion	EK–600SB–IN
VAX 6000 Platform Service Manual	EK–600EA–MG
Options and Upgrades	
VAX 6000: XMI Conversion Manual	EK–650EB–UP
VAX 6000: Installing MS65A Memories	EK–MS65A–UP
VAX 6000: Installing the H7236-A Battery Backup Option	EK–60BBA–IN
VAX 6000: Installing the VAXBI Option	EK–60BIA–IN
Model 600	
VAX 6000 Model 600 Mini-Reference	EK–660EA–HR
VAX 6000 Model 600 Service Manual	EK–660EA–MG
VAX 6000 Model 600 System Technical User's Guide	EK–660EA–TM
VAX 6000: Installing Model 600 Processors	EK–660EA–UP

Associated Documents

Table 2 lists other documents that you may find useful.

Table 2: Associated Documents

Title	Order Number
System Hardware Options	
VAXBI Expander Cabinet Installation Guide	EK–VBIEA–IN
VAXBI Options Handbook	EB–32255–46
System I/O Options	
CIBCA User Guide	EK–CIBCA–UG
CIXCD Interface User Guide	EK–CIXCD–UG
DEC LANcontroller 200 Installation Guide	EK–DEBNI–IN
DEC LANcontroller 400 Installation Guide	EK–DEMNA–IN
DSSI VAXcluster Installation Guide	EK–DVCLU–IN
InfoServer Installation Guide	EK–DIS1K–IN
KDB50 Disk Controller User's Guide	EK–KDB50–UG
KDM70 Controller User Guide	EK–KDM70–UG
KFMSA Module Installation and User Manual	EK–KFMSA–IM
KFMSA Module Service Guide	EK–KFMSA–SV
RRD42 Disc Drive Owner's Manual	EK–RRD42–OM
RA90/RA92 Disk Drive User Guide	EK–ORA90–UG
RF31/RF72 Integrated Storage Element Installation Manual for BA200-Series Enclosures	EK–RF72D–IM
RF31/RF72 Integrated Storage Element User Guide	EK–RF72D–UF
RF31/RF72 Integrated Storage Element Service Guide	EK–RF72D–SV
SA70 Enclosure User Guide	EK–SA70E–UG
SF2xx Storage Array Installation Guide	EK–SF200–IG
SF7x Storage Enclosure and SF2xx Storage Array Cabinet Service Guide	EK–SF72S–SG
TF85 Cartridge Tape Subsystem Owner's Manual	EK–OTF85–OM
TF857 Magazine Tape Subsystem Service Manual	EK–TF857–OM
VAX 6000/SF2xx Embedded Storage Installation Guide	EK–EMBED–IN

Table 2 (Cont.): Associated Documents

Title	Order Number
Operating System Manuals	
Guide to Maintaining a VMS System	AA–LA34B–TE
Guide to Setting Up a VMS System	AA–LA25A–TE
Introduction to VMS System Management	AA–LA24A–TE
ULTRIX–32 Guide to System Exercisers	AA–ME96B–TE
VMS Networking Manual	AA–LA48A–TE
VMS System Manager's Manual	AA–LA00B–TE
VMS Upgrade and Installation Supplement: VAX 6000 Series	AA–LB36C–TE
VMS Version 5.5 Upgrade and Installation Manual	AA–NG61D–TE
VAXclusters and Networking	
DECbridge 500 Installation Guide	EK–DEFEB–IN
DEMFA Installation Guide	EK–DEMFA–IN
Fiber Distributed Data Interface Description	EK–DFSLD–SD
Guidelines for VAXcluster System Configurations	EK–VAXCS–CG
H4000 Digital Ethernet Transceiver Installation Manual	EK–H4000–IN
HSC Installation Manual	EK–HSCMN–IN
VAXcluster Principles	EK–VAXCP–TM
VMS VAXcluster Manual	AA–LA27B–TE

Table 2 (Cont.): Associated Documents

Title	Order Number
Peripherals	
Installing and Using the VT420 Video Terminal	EK–VT420–UG
RV20 Optical Disk Owner's Manual	EK–ORV20–OM
SC008 Star Coupler User's Guide	EK–SC008–UG
TA78 Magnetic Tape Drive User's Guide	EK–OTA78–UG
TA90 Magnetic Tape Subsystem Owner's Manual	EK–OTA90–OM
TK70 Streaming Tape Drive Owner's Manual	EK–OTK70–OM
TU81/TA81 and TU/81 PLUS Subsystem User's Guide	EK–TUA81–UG
VAX Manuals	
VAX Architecture Reference Manual	EY–3459E–DP
VAX Systems Hardware Handbook — VAXBI Systems	EB–31692–46

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

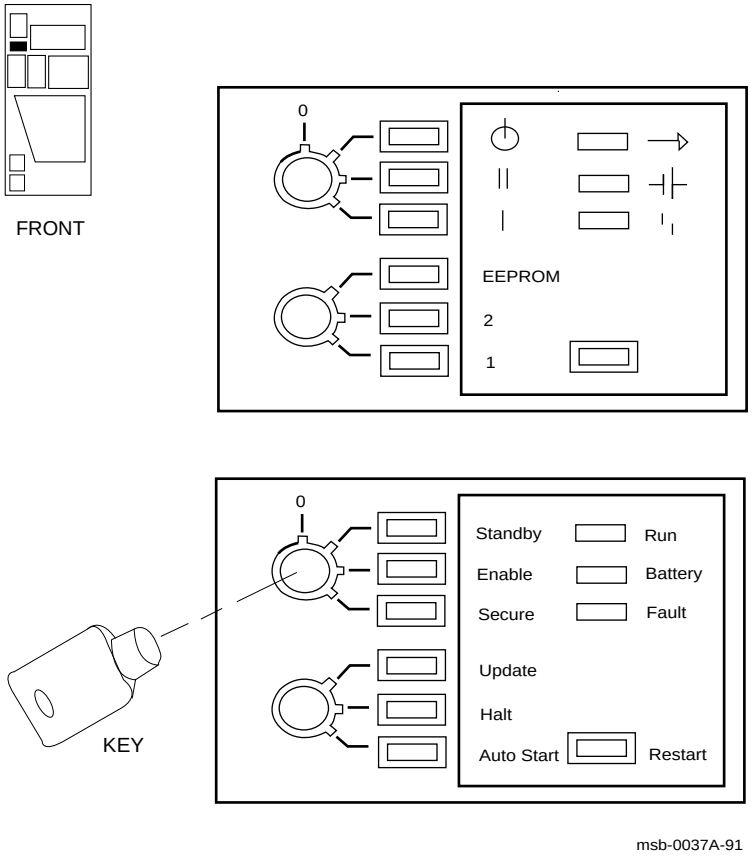


Table 1–1: Upper Key Switch

Position	Effect	Light Color
O (Off)	Removes all power, except to the battery backup charger and optional storage.	No light
Standby	Supplies power to XMI backplane, blowers, and in-cabinet console load device.	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. When switch at Secure, the system performs an automatic restart, regardless of the setting of the lower key switch.	Green

Table 1–2: Lower Key Switch

Position	Effect	Light Color
Update	Enables writing to CPUs and adapters. Halts boot processor in console mode on power-up or when Restart button is pressed. Used for updating parameters stored in EEPROMs (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 reboot. 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 charged to 98% of full capacity or BBU is supplying power to the load.
		Flashing 1 x/sec	Battery backup unit is charging.
		Flashing 10 x/sec	Battery backup unit requires service.
		Off	System does not have a battery backup unit.
Fault	Red	On	Self-test is in progress. If light does not turn off, system has a hardware fault. See *VAX 6000 Series Owner's Manual* 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 /FILENAME:xyz /DSSI_NODE:n /PORT:x	Initializes the system, causing a self-test, and begins the boot program.
CLEAR EXCEPTION	Cleans up error state in XBER, XBEER, and CPU-specific 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 reset, including self-test.
REPEAT	Executes the command passed as its argument.
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.
SHOW ALL	Displays the current value of parameters set.
SHOW BOOT	Displays all boot commands and nicknames that have been saved using SET BOOT.

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

Command and Qualifiers	Function
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 DSSI	Displays DSSI bus numbers, node numbers, and unit numbers.
SHOW ETHERNET	Displays Ethernet hardware addresses for all Ethernet adapters on the system and FDDI hardware addresses for all FDDI adapters.
SHOW FIELD	Displays saved boot commands, console terminal parameters, console language mode, memory configuration, type of power system, and system serial number.
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 another processor.
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 the 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

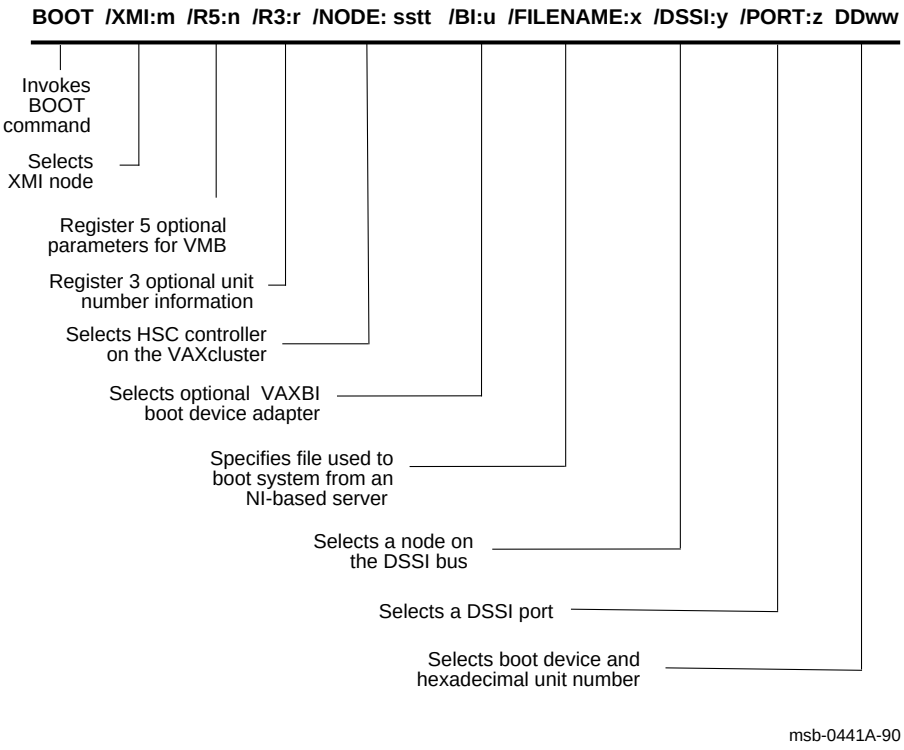


Table 1–7: BOOT Command Qualifiers

Qualifier	Function
/X[MI]:number	Specifies the XMI node number of the node that connects the boot device.
/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.
/B[I]:number	Specifies a VAXBI node that connects the boot device. The /XMI qualifier must have selected a node containing a DWMBB/A.
/FILE[NAME]:file	Specifies the file name used to boot from an Ethernet-based server. The file name can be 1 to 16 characters in length.
/D[SSI_NODE]:number	Specifies the DSSI node that provides access to the boot device. The /XMI qualifier must have selected a node containing a KFMSA adapter.
/PO[RT]:number	Specifies DSSI port 1 or 2 on the KFMSA adapter.

Table 1–8: Sample BOOT Commands

Boot Procedure	BOOT Command
Boot from in-cabinet console load device	BOOT CSA1
Boot VAX/DS from an in-cabinet console load device	BOOT /R5:10 CSA1
Boot from local RA disk	BOOT /XMI:m DUww
Boot from local RF disk	BOOT /XMI:m /DSSI_NODE:y /PORT:z DIww
Boot from HSC disk	BOOT /XMI:m /R5:v/NODE:sstt DUww
Boot from a DSSI TF tape	BOOT /XMI:m /DSSI_NODE:y /PORT:z MIww
Boot from an Ethernet-based CD server	BOOT /XMI:m /FILENAME:ISL_LVAX_n EX0
Boot over FDDI from a CD server	BOOT /XMI:m /FILENAME:ISL_LVAX_n FX0
Boot VAX/DS from an Ethernet-based CD server	BOOT /XMI:m/FILENAME:ISL_LVAX_x[1] /R5:10 EX0
Boot over the Ethernet from a VAXBI device	BOOT /XMI:m /BI:x ET0
Boot VAX/DS from disk	BOOT /XMI:m /R5:10 DUww
Conversational boot	BOOT /XMI:m /R5:1 DUww
Boot from VMS shadow set	BOOT /XMI:m /R3:w /NODE:sstt DUww

[1]Where *x* is a letter that indicates the version.

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 the 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 lists the console error messages that appear when the processor halts and the console gains control. Most messages are followed by:

- PC = xxxxxxxx — program counter = address at which the processor halted or the exception occurred
- PSL = xxxxxxxx — processor status longword = contents of the register
- –SP = xxxxxxxx — –SP is one of the following:
  ESP executive stack pointer
  ISP interrupt stack pointer
  KSP kernel stack pointer
  SSP supervisor stack pointer
  USP user stack pointer

Table 1–12 lists standard console error messages.

Table 1–11: Console Error Messages Indicating Halt

Error Message	Meaning
?0002 External halt (CTRL/P, break, or external halt).	CTRL/P or STOP command.
?0003 Power-up halt.	System has powered up, had a system reset, or an XMI node reset.
?0004 Interrupt stack not valid during exception processing.	Interrupt stack pointer contained an invalid address.
?0005 Machine check occurred during exception processing.	A machine check occurred while handling another error condition.
?0006 Halt instruction executed in kernel mode.	The CPU executed a Halt instruction.
?0007 SCB vector bits <1:0> = 11.	An interrupt or exception vector in the System Control Block contained an invalid address.
?0008 SCB vector bits <1:0> = 10.	An interrupt or exception vector in the System Control Block contained an invalid address.
?000A CHMx executed while on interrupt stack.	A change-mode instruction was issued while executing on the interrupt stack.

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

Error Message	Meaning
?0010 ACV/TNV occurred during machine check processing.	An access violation or translation-not-valid error occurred while handling another error condition.
?0011 ACV/TNV occurred during kernel-stack-not-valid processing.	An access violation or translation-not-valid error occurred while handling another error condition.
?0012 Machine check occurred during machine check processing.	A machine check occurred while processing a machine check.
?0013 Machine check occurred during kernel-stack-not-valid processing.	A machine check occurred while handling another error condition.
?0019 PSL <26:24>= 101 during interrupt or exception.	An exception or interrupt occurred while on the interrupt stack but not in kernel mode.
?001A PSL <26:24>= 110 during interrupt or exception.	An exception or interrupt occurred while on the interrupt stack but not in kernel mode.
?001B PSL <26:24>= 111 during interrupt or exception.	An exception or interrupt occurred while on the interrupt stack but not in kernel mode.
?001D 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.
?001E 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.
?001F 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
?0020 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.
?0021 Illegal command.	The command was not recognized, contained the wrong number of parameters, or contained unrecognized or inappropriate qualifiers.
?0022 Illegal address.	The specified address was recognized as being invalid, for example, a general purpose register number greater than 15.
?0023 Value is too large.	A parameter or qualifier value contained too many digits.
?0024 Conflicting qualifiers.	A command specified recognized qualifiers that are illegal in combination.
?0025 Checksum did not match.	The checksum calculated for a block of X command data did not match the checksum received.
?0026 Halted.	The processor is currently halted.
?0027 Item was not found.	The item requested in a FIND command could not be found.
?0028 Timeout while waiting for characters.	The X command failed to receive a full block of data within the timeout period.
?0029 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.
?002A Unexpected machine check or interrupt.	A valid operation within the console caused a machine check or interrupt.
?002B Command is not implemented.	The command is not implemented by this console.
?002C Unexpected exception.	An attempt was made to examine either a nonexistent IPR or an unimplemented register in RSSC address range (20140000—20140800).

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

Error Message	Meaning
?002D 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.
?002E Specified node is not an I/O adapter.	The referenced node is incapable of performing I/O or did not pass its self-test.
?0030 Write to Z command target has timed out.	The target node of the Z command is not responding.
?0031 Z connection terminated by ^P.	A CTRL/P was typed on the keyboard to terminate a Z command.
?0032 Your node is already part of a Z connection.	You cannot issue a Z command while executing a Z command.
?0033 Z connection successfully started.	You have requested a Z connection to a valid node.
?0034 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.
?0036 Command too long.	The command length exceeds 80 characters.
?0037 Bad explicit interleave list — 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.
?0039 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.
?003B Error encountered during I/O operation.	An I/O adapter returned an error status while the console boot primitive was performing I/O.
?003C 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.

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

Error Message	Meaning
?003D Error initializing I/O device.	A console boot primitive needed to perform I/O, but could not initialize the I/O adapter.
?003E 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.
?0040 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.
?0041 Specified node is not a bus adapter.	A command to access a VAXBI node specified an XMI node that was not a bus adapter.
?0042 Invalid terminal speed.	The SET TERMINAL command specified an unsupported baud rate.
?0043 Unable to initialize node.	The INITIALIZE command failed to reset the specified node.
?0044 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.
?0045 Unable to stop node.	The STOP command failed to halt the specified node.
?0046 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.
?0047 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.
?0049 Memory cannot be initialized.	The specified operation was attempted and prevented.

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

Error Message	Meaning
?004A 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.
?004B Internal logic error in console.	The console encountered a theoretically impossible condition.
?004C 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.
?004D Invalid node for new primary.	The SET CPU command failed when attempting to make the specified node the primary processor.
?004E Specified node is not a processor.	The specified node is not a processor, as required by the command.
?004F System serial number has not been initialized.	No CPU in the system contains a valid system serial number.
?0050 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.
?0051 Secondary processor returned bad response message.	A secondary processor returned an unintelligible response to a request made by the console on the primary processor.
?0052 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.
?0053 EEPROM header is corrupted.	The EEPROM header has been corrupted. The EEPROM must be restored from the TK tape drive.
?0054 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.
?0055 Failed to locate EEPROM area.	The EEPROM did not contain a set of data required by the console. The EEPROM may be corrupted.

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

Error Message	Meaning
?0056 Console parameters on secondary processor do not match primary.	The console parameters are not the same for all processors.
?0057 EEPROM area checksum error.	A portion of the EEPROM is corrupted. It may be necessary to reload the EEPROM from the TK tape drive.
?0058 Saved boot specifications on secondary processor do not match primary.	The saved boot specifications are not the same for all processors.
?0059 Invalid unit number.	A BOOT or SET BOOT command specified a unit number that is not a valid hexadecimal number between 0 and FF.
?005A System serial number mismatch. Secondary processor has xxxxxxxx.	The indicated serial number of a secondary processor does not match that of the primary.
?005B 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.
?005C No HELP is available.	The HELP command is not supported when the console language is set to International.
?005D No such boot spec found.	The specified boot specification was not found in the EEPROM.
?005E Saved boot spec table full.	The maximum number of saved boot specifications has already been stored.
?005F EEPROM header version mismatch.	Processors have different versions of EEPROMs.
?0061 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.
?0062 Illegal node number.	The specified node number is invalid.
?0063 Unable to locate console tape device.	The console could not locate the I/O adapter that controls the TK tape.
?0064 Operation only applies to secondary processors.	The command can only be directed at a secondary processor.
?0065 Operation not allowed from secondary processor.	A secondary processor cannot perform this operation.

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

Error Message	Meaning
?0066 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.
?0067 Read of EEPROM image from tape failed.	The EEPROM image was not successfully read from tape.
?0068 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.
?0069 EEPROM not changed.	The EEPROM contents were not changed.
?006A EEPROM changed successfully.	The EEPROM contents were successfully patched or restored.
?006B Error changing EEPROM.	An error occurred in writing to the EEPROM. The EEPROM contents may be corrupted.
?006C EEPROM saved to tape successfully.	The EEPROM contents were successfully written to the TK tape.
?006D EEPROM not saved to tape.	The EEPROM contents were not completely written to the TK tape.
?006E EEPROM Revision = *x.xx/y.yy*.	The EEPROM contents are at revision *x.xx* with revision *y.yy* patches.
?006F Major revision mismatch between tape image and EEPROM.	The major revision of tape and EEPROM do not match. The requested operation cannot be performed.
?0070 Tape image Revision = *x.xx/y.yy*.	The EEPROM image on the TK tape is at revision *x.xx* with revision *y.yy* patches.
?0073 System serial number updated.	The EEPROM has been updated with the correct system serial number.
?0074 System serial number not updated.	The EEPROM has not been changed.
?0075 /CONSOLE_LIMIT value too small for proper operation. Value ignored.	No change has been made.
?0076 Error writing to tape. Tape may be write-locked.	Tape has not been written. Check to see if tape is write-locked.

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

Error Message	Meaning
?0077 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.
?007C I/O adapter configuration error at node *n*	The I/O adapter at node *n* is configured improperly.
?0083 Loading system software.[1]	The console is attempting to load the operating system in response to a BOOT command, power-up, or restart failure.
?0084 Failure.[1]	An operation did not complete successfully. Should be issued with another message to clarify failure.
?0085 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.
?00A0 Initializing system.[1]	The console is resetting the system in response to a BOOT command.
?00A1 Now updating the EEPROM of node *n*[1]	The console is updating the EEPROM.
?00A6 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.
?00A7 RCSR <WD> is set. Local CCA must be built.[1]	When the <WD> bit is set, writes to memory are disabled.
?00A8 Bootstrap failed due to previous error.[1]	The previous attempt to bootstrap the system failed.
?00A9 Restart failed due to previous error.[1]	The previous attempt to restart the system failed.
Node *n: ?xxxx*	Error message *?xxxx* was generated on secondary processor *n* and was passed to the primary processor to be displayed.
?0104 Filename format error.	Period and semicolon characters are improperly used within the filename specified for a MOP boot.

[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
?0105 Illegal character(s) in filename.	For filename specified in a MOP boot.
?0106 Filename cannot contain nested blanks or tabs.	For filename specified in a MOP boot.
?0107 Filename can be no longer than 16 characters.	For filename specified in a MOP boot.
?011E Uncorrectable memory errors discovered - long memory test must be performed on node *n*	Memory array in node *n* contains an uncorrectable error. The console must perform a full test to locate all the failing locations.
?0120 Unsupported memory module found, will not be configured.	One or more MS62A memory modules are installed but will not be used. Only MS65A memory modules are compatible with Model 600 CPUs.
?0121 Patch command no longer implemented—use the diagnostic utility EVUCA.	An invalid PATCH command was issued; use the EVUCA program to update the EEPROM.
?0201 One or more power-up tests have been bypassed.	A test normally run by the processor at power-up has been bypassed.
?0203 Hardware compatibility group mismatch—secondary/primary: *x/y*.	Hardware version mismatch between the primary CPU and an indicated secondary CPU.
?0205 Error locating ROM boot code, run diagnostics.	The console had a problem reading the CPU's ROM code.
?0206 EEPROM in error or contains unsupported PCS, processor disabled.	The EEPROM image is the wrong version or is faulty. Use the EVUCA program to upgrade the EEPROM for the indicated CPU.

Boot and Status Error Messages

The following lists show the status and error messages for Ethernet boots, local disk and tape boots, and cluster boots. Status messages are shown in the order they would appear after the boot command is issued. Listed after each status message are the error messages that could appear during each boot subprocess.

Ethernet Boot Messages

1. [Start boot]

   ?002E Specified node is not an I/O adapter
   ?0100 Specified adapter failed self-test
   ?010B Illegal adapter specified for NI boot

2. * Initializing adapter

   ?0119 Failure to initialize specified adapter

3. * Specified adapter initialized successfully

4. * "Request Program" MOP message sent—waiting for service from remote node

   ?0113 No traffic was detected on the Ethernet—aborting boot procedure
   ?0115 Aborting boot process—adapter failed attempting to execute port command
   ?011F Aborting boot process—adapter failed attempting to execute boot command

5. * Still waiting for assistance—reissuing "Request Program" message

6. * Remote service link established

7. * Reading boot image from remote node

   ?010F Failed to receive image from remote server

8. * Passing control to transfer address

Local Disk Boot Messages

1. [Start Boot]

   ?002E Specified node is not an I/O adapter
   ?0100 Specified adapter failed self-test
   ?010A Illegal adapter specified for disk boot

2. * Initializing adapter

   ?0119 Failure to initialize specified adapter

3. * Specified adapter initialized successfully

4. * Connecting to boot disk *or*
   * Reading bootblock from disk

   ?0102 Controller error detected—aborting
   ?0103 Drive error detected—aborting

?010E Specified unit offline — No media mounted or disabled via RUN/STOP switch setting
?0114 Serious exception reported—aborting
?0116 Specified unit is inoperative
?0117 Specified unit offline
?0118 Specified unit offline—Unit unknown, online to another controller or port disabled via A,B switches

5. * Passing control to transfer address

Local Tape Boot Messages

1. [Start boot]

?002E Specified node is not an I/O adapter
?0100 Specified adapter failed self-test
?010C Illegal adapter specified for tape use

2. * Initializing adapter

?0119 Failure to initialize specified adapter

3. * Specified adapter initialized successfully

4. * Connecting to tape *or*
   * Reading bootblock from tape *or*
   * Rewinding tape

?0101 BVP port error reported—aborting
?0102 Controller error detected—aborting
?0103 Drive error detected—aborting
?010E Specified unit offline—No media mounted or disabled via RUN/STOP switch setting
?0114 Serious exception reported—aborting
?0116 Specified unit is inoperative
?0117 Specified unit offline
?0118 Specified unit offline—Unit unknown, online to another controller or port disabled via A,B switches

5. * Passing control to transfer address

CI and DSSI Boot Messages

1. [Start boot]

?002E Specified node is not an I/O adapter
?0109 Illegal adapter specified for CI boot
?011A Illegal adapter specified for DSSI boot

2. * Initializing adapter

   ?0119 Failure to initialize specified adapter

3. * Specified adapter initialized successfully

4. * Connecting to storage controller

5. * Previous operation failed—retrying CI boot

6. * Previous operation failed—retrying DSSI boot

7. * Port received a "no path" error—retrying the init sequence

   ?0110 Port received a "no path" error after 6 retries—aborting the boot process

8. * Connecting to MSCP server layer

9. * Previous operation failed—retrying CI boot

10. * Connecting to boot disk *or*
    * Connecting to shadow unit—will fail over to physical after 6 attempts.

    ?0102 Controller error detected—aborting
    ?0103 Drive error detected—aborting
    ?010E Specified unit offline—No media mounted or disabled via RUN/STOP switch setting
    ?0114 Serious exception reported—aborting
    ?0116 Specified unit is inoperative
    ?0117 Specified unit offline
    ?0118 Specified unit offline—Unit unknown, online to another controller or port disabled via A,B switches

11. * Failure to connect to shadow unit—retrying on physical unit

12. * Reading bootblock from disk

    ?0102 Controller error detected—aborting
    ?0103 Drive error detected—aborting
    ?010E Specified unit offline—No media mounted or disabled via RUN/STOP switch setting
    ?0114 Serious exception reported—aborting
    ?0116 Specified unit is inoperative
    ?0117 Specified unit offline
    ?0118 Specified unit offline—Unit unknown, online to another controller or port disabled via A,B switches

13. * Passing control to transfer address

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 0123456789 012345#  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
    +   .   +   .   +   +   +   +   .   .   +   +   -   +       STF    ❹
    .   .   .   .   .   .   .   .   .   .   E   E   E   B       BPD    ❺
    .   .   .   .   .   .   .   .   .   .   +   +   -   -       ETF    ❻
    .   .   .   .   .   .   .   .   .   .   E   B   E   E       BPD    ❺

    .   .   .   .   A4  A3  A2  A1  .   .   .   .   .   .       ILV    ❼
    .   .   .   .   64  64  64  64  .   .   .   .   .   .     256 Mb   ❽

Console = V1.00   RBDs = V1.00  EEPROM = 1.00/1.01  SN = SG01234567
              ❾              ❿              ⓫

>>>
```

❶ The first line in Example 2–1 shows that the CPU in slot 1 passed all testing. If the final # sign is missing, the last number shown is the number of the failing test. This line of numbers is displayed only by the processor in node 1 — and only when this processor undergoes power-up or a system reset. This processor is not always the boot processor.

❷ The NODE # line lists the node numbers on the XMI bus. The nodes on this line are numbered in hexadecimal and reflect the position of the XMI slots as you view the XMI from the front of the cabinet through the clear card cage door.

❸ The TYP line in the printout indicates the type of module at each node:

A = I/O adapter
P = processor
M = memory module

❹ The STF line shows the results of self-test. This information is taken from the self-test fail bit in the XBER register of each module. The entries are:

\+ (pass)
– (fail)
o (does not apply)

❺ The BPD line indicates boot processor designation.

The results on the BPD line indicate:

B = Boot processor
E = Processors eligible to become boot processor
D = Processors ineligible to become boot processor

This BPD line is printed twice. After the first determination of the boot processor, the processors go through an extended test. Since it is possible for a processor to pass self-test (at line STF) and fail the extended test (at ETF), the processors again determine the boot processor following the extended test.

❻ During the extended test (ETF) all processors run additional CPU tests involving memory. Results printed at this ETF line indicate:

- Two processors passed the extended test (+)
- Two processors failed the extended test (–)

❼ This ILV line contains a memory interleave value (ILV) for each memory. If you have more than one interleave set, each set is indicated by a different letter.

❽ The line after the ILV line displays the size of each memory module configured in the system and gives the total Mbytes of system memory. In Example 2–1 the total is 256 Mbytes.

❾ Console and RBD information indicates the version of read-only memory that is installed on the processors in this system. Each processor has a console ROM and an RBD ROM; each ROM has its own version. In Example 2–1 all processors have version V1.00 ROM resident. All processors should run with the same level of ROM. If your processors have mixed levels of ROM, the ROM level of the primary processor is displayed here, and you receive an error message that your processors have different ROM levels.

⓾ The EEPROM information gives the boot processor's version of EEPROM and the patch level. In Example 2–1 the first number, 1.00, gives the version of the contents of the EEPROM, and the second number, 1.01, is the console patch level. If you run processors whose EEPROMs do not match, you will receive an error message.

⓫ SN gives the system serial number. The system serial number is also on the cabinet.

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

Module	XMI Node Number	Module Type
KA66A	1	Processor; boot processor after first level of self-test, fails extended test.
KA66A	2	Processor; fails first level of self-test and extended test.
KA66A	3	Processor; operating as boot processor.
KA66A	4	Processor; passes first level of self-test and extended test.
MS65A	7	Memory (64 Mbytes); interleaved with memories at other nodes.
MS65A	8	Memory (64 Mbytes); interleaved with memories at other nodes.
MS65A	9	Memory (64 Mbytes); interleaved with memories at other nodes.
MS65A	A	Memory (64 Mbytes); interleaved with memories at other nodes.
CIXCD	C	I/O adapter; passes self-test.
DEMNA	E	I/O adapter; passes self-test.

Example 2-2 shows a self-test display that contains an additional line when an optional VAXBI adapter (DWMBB) is part of the system configuration. The XBI line provides information on the node numbers and self-test status for modules in the VAXBI card cages, which are connected to the XMI through a DWMBB.

Example 2–2: Sample Self-Test Results with VAXBI Adapter

```
#123456789 0123456789 0123456789 0123456789 012345#

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

    A   .   A   .   M   M   M   M   .   .   P   P   P   P       TYP     ❶
    o   .   +   .   +   +   +   +   .   .   +   +   -   +       STF     ❷
    .   .   .   .   .   .   .   .   .   .   E   E   E   B       BPD
    .   .   .   .   .   .   .   .   .   .   +   +   -   -       ETF
    .   .   .   .   .   .   .   .   .   .   E   B   E   E       BPD

.   .   .   .   .   .   .   .   .   +   .   +   -   +   +   .   XBI E +❸

    .   .   .   .   A4  A3  A2  A1  .   .   .   .   .   .       ILV
    .   .   .   .   64  64  64  64  .   .   .   .   .   .     256 Mb

Console = V1.00   RBDs = V1.00    EEPROM = 1.00/1.01     SN = SG01234567

>>>
```

The system configuration shown in Example 2–2 contains a DWMBB/A module in XMI slot E.

❶ The TYP line in this printout indicates that the adapters in this configuration are in XMI slots C and E.

❷ Because the DWMBB does not have a module-resident self-test, its entry for the STF line will always be "o".

❸ The test results for the DWMBB/A and the DWMBB/B modules are indicated on the XBI line, at the far right. In this example, the DWMBB modules have passed self-test (**XBI E +**). The results of the VAXBI I/O adapter self-tests are shown in columns 1 through F, which stand for the VAXBI node numbers; in this configuration, node numbers 1, 2, 3, 4, and 6 are used. The adapter at node 3 failed its self-test.

Table 2–2 lists each module's LED status indicating self-test passed or self-test failed.

Table 2–2: Module LEDs After Self-Test

Module	Self-Test Passed	Self-Test Failed
Boot processor	Yellow ON Top two red ON and bottom red OFF	Yellow OFF Some red ON [1]
Secondary processor(s)	Yellow ON Top two red ON and bottom red ON	Yellow OFF Some red ON [1]
Memory	Yellow ON Green ON	Yellow ON [2] Green ON
VAXBI adapter	Yellow ON	Yellow OFF

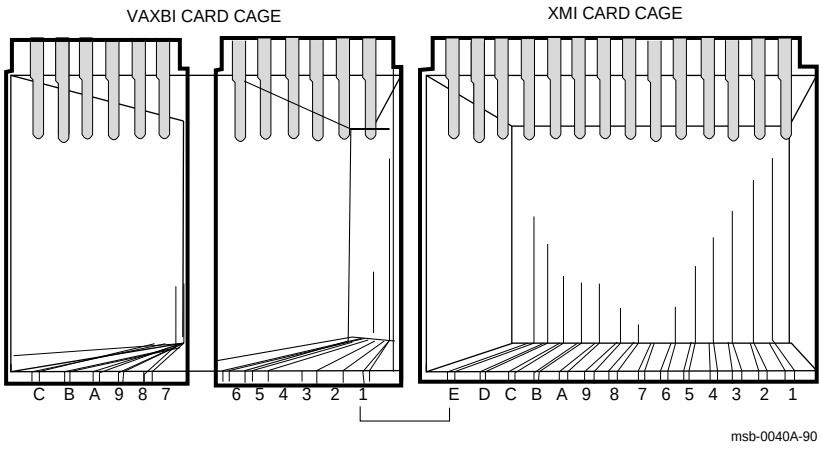
[1] Processor modules have eight red LEDs that display the number of the test that failed. Refer to the *VAX 6000 Model 600 Service Manual* for more information.

[2] The yellow indicator on the memory module is used to indicate *only* that self-test has completed.

Chapter 3
Address Space

The design of the hardware for the system bus (the XMI) and for the optional VAXBI bus affects addressing. The XMI card cage has its 14 slots permanently assigned to specific address locations. For the Model 600, no modules that require I/O cables can be installed in the middle four slots (slots 6 through 9). The VAXBI bus consists of two VAXBI card cages that are physically fastened together and logically connected as one 12-slot VAXBI bus. For more information on VAXBI node addressing, see Section 3.3.

Figure 3–1: VAX 6000 Model 600 Slot Numbers



3.1 Physical Address Space

The KA66A CPU generates a 32-bit address that corresponds to 4 gigabytes of physical address space. However, the CPU can run in 30-bit mode as well. Figure 3–2 shows the layout of both memory and I/O space. I/O space occupies the last one-eighth (512 Mbytes) of the physical address space and can be distinguished from memory space by the fact that bits <31:29> of the physical address are all ones.

The translation from a 30-bit address to a 32-bit address is accomplished by sign-extending physical address <29> to physical address <31:29>. In this mode the programmer sees a 1-Gbyte address space, split evenly between memory and I/O space. A 30-bit address is mapped to the actual 32-bit physical address as shown in Table 3–1.

Figure 3–2: Physical Address Space Layout

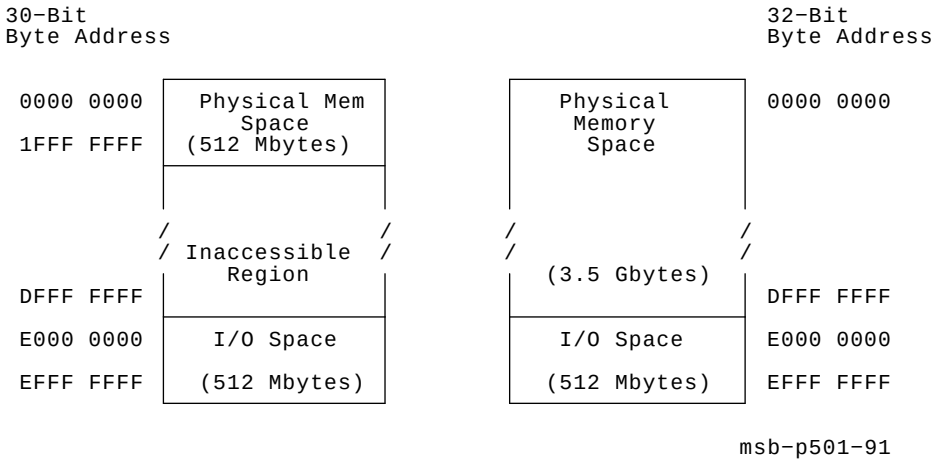


Table 3–1: 30-Bit Mapping of Program Addresses to 32-Bit Hardware Addresses

Program Address	Hardware Address
00000000–1FFFFFFF	00000000–1FFFFFFF
20000000–3FFFFFFF	E0000000–FFFFFFFF

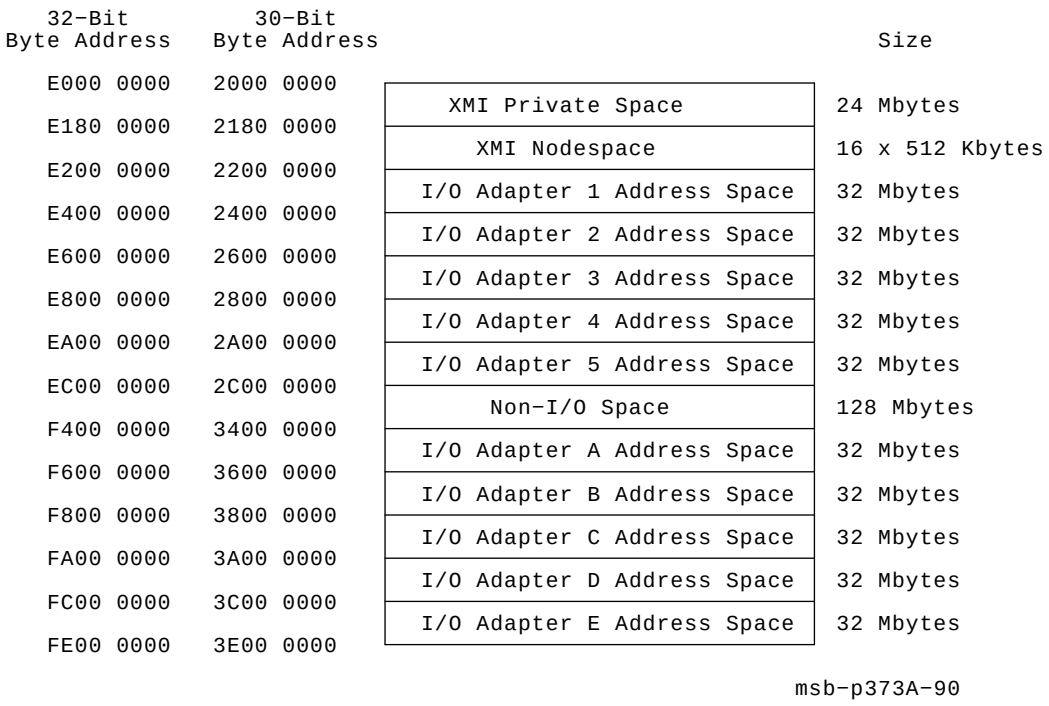
During power-up, microcode configures the CPU to generate 30-bit physical addresses. Operating system initialization code can reconfigure the CPU to generate either 30-bit or 32-bit physical addresses by writing to the MODE bit <0> in the Physical Address Mode Register (IPR231). For full details on physical address space, see the *VAX 6000 Model 600 System Technical User's Guide* and the *VAX 6000 Platform Technical User's Guide*.

Register addresses for a particular device in a system are found by adding an offset to the base address for that particular 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

XMI I/O space is divided into private space, nodespace, and ten I/O adapter address space regions.

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



msb–p373A–90

XMI Private Space

References to XMI private space are serviced by resources local to a node, such as local device CSRs and boot ROM. The references are not broadcast on the XMI. XMI private space is a 24-Mbyte address region located from E000 0000 to E17F FFFF.

XMI Nodespace

The VAX 6000 platform XMI nodespace is a collection of 16 512-Kbyte regions located from E180 0000 to E1FF FFFF. Nodes 0 and F are not implemented. Each XMI node is allocated one of the 512-Kbyte regions for its control and status registers. The starting address of the 512-Kbyte region associated with a given node is computed as follows:

E180 0000 + (Node ID x 80000)

Table 3–2: XMI Nodespace Addresses

Slot	Node	Nodespace	I/O Window Space
1	1	E188 0000 – E18F FFFF	E200 0000 – E3FF FFFF
2	2	E190 0000 – E197 FFFF	E400 0000 – E5FF FFFF
3	3	E198 0000 – E19F FFFF	E600 0000 – E7FF FFFF
4	4	E1A0 0000 – E1A7 FFFF	E800 0000 – E9FF FFFF
5	5	E1A8 0000 – E1AF FFFF	EA00 0000 – EBFF FFFF
6	6	E1B0 0000 – E1B7 FFFF	N/A[1]
7	7	E1B8 0000 – E1BF FFFF	N/A[1]
8	8	E1C0 0000 – E1C7 FFFF	N/A[1]
9	9	E1C8 0000 – E1CF FFFF	N/A[1]
10	A	E1D0 0000 – E1D7 FFFF	F400 0000 – F5FF FFFF
11	B	E1D8 0000 – E1DF FFFF	F600 0000 – F7FF FFFF
12	C	E1E0 0000 – E1E7 FFFF	F800 0000 – F9FF FFFF
13	D	E1E8 0000 – E1EF FFFF	FA00 0000 – FBFF FFFF
14	E	E1F0 0000 – E1F7 FFFF	FC00 0000 – FDFF FFFF

[1]Slots in the center of the XMI card cage have no I/O connectors because of the daughter card's presence.

3.2 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+  KA66A   (8087) 0003
  2+  KA66A   (8087) 0003
  6+  MS65A   (4001) 0084
  7+  MS65A   (4001) 0084
  8+  MS65A   (4001) 0084
  9+  MS65A   (4001) 0084
  B+  DEMNA   (0C03) 0003
  C+  KDM70   (0C22) 0001
  E+  DWMBB/A (2002) 0002

  XBI E
  1+  DWMBB/B (2107) 0007
  4+  DMB32   (0109) 210B
  6+  TBK70   (410B) 0307
```

Assume that you want to examine the Bus Error Register (XBER) of the DEMNA module in slot 11, which is XMI node B. From Table 3–2, XMI Nodespace Addresses, you can see that the nodespace base address for the XMI module at node B is E1D8 0000. From Table 6–2, XMI Required Registers, 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 E1D80004
```

3.3 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 DWMBB/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+  KA66A   (8087) 0003
  2+  KA66A   (8087) 0003
  6+  MS65A   (4001) 0084
  7+  MS65A   (4001) 0084
  8+  MS65A   (4001) 0084
  9+  MS65A   (4001) 0084
  B+  DEMNA   (0C03) 0003
  C+  KDM70   (0C22) 0001
  E+  DWMBB/A (2002) 0002

  XBI E
  1+  DWMBB/B (2107) 0007
  4+  DMB32   (0109) 210B
  6+  TBK70   (410B) 0307
```

Assume that you want to examine the Device Register (DTYPE) for the DMB32, which is node 4 in the VAXBI channel shown above (XBI E).

To get the address for the DMB32 Device Register (DTYPE), do the following:

1. From Table 3–2, XMI Nodespace Addresses, find XMI node E and take the first two digits for that node's window space (FC).

2. From Table 3–3 find VAXBI node 4 and in column 2 you can see that the starting address for VAXBI node 4 is xx00 8000.
3. Combine this second number with the two digits. You now have the adapter's base address (FC00 8000) in VAXBI address space, indicated by lowercase bb.
4. From Table 3–4, VAXBI Registers, you can see that the VAXBI Device Register (DTYPE) is at bb + 00, which is FC00 8000.

The Device Register for the DMB32 would be examined by:

```
>>>  E/L/P FC008000
```

Table 3–3: 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 3–4: 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
KA66A CPU Module Registers

The KA66A module 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, Bits, and Fields

Type	Description
MBZ	Must be zero
0	Initialized to logic level zero
1	Initialized to logic level one
X	Initialized to either logic level
RO	Read only
R/W	Read/write
R/Cleared on W	Read/cleared on write
R/W1C	Read/cleared by writing a one
WO	Write only

The following rules govern overwriting of information in the error registers:

- If no error information is in the error registers, they are written on the first hard or soft error.
- If soft error information is being latched, the error registers are not changed on subsequent soft errors.
- If soft error information is being latched, the error registers are overwritten by a hard error.
- If hard error information is being latched, the information is not changed on subsequent errors.

4.1 KA66A Internal Processor Registers

Table 4–2: KA66A Internal Processor Registers

Address Dec. (Hex)	Register	Mnemonic	Type[1]	Class[2]
0 (0)	Kernel Stack Pointer	KSP	R/W	1
1 (1)	Executive Stack Pointer	ESP	R/W	1
2 (2)	Supervisor Stack Pointer	SSP	R/W	1
3 (3)	User Stack Pointer	USP	R/W	1
4 (4)	Interrupt Stack Pointer	ISP	R/W	1
8 (8)	P0 Base	P0BR	R/W	1
9 (9)	P0 Length	P0LR	R/W	1
10 (A)	P1 Base	P1BR	R/W	1
11 (B)	P1 Length	P1LR	R/W	1
12 (C)	System Base	SBR	R/W	1
13 (D)	System Length	SLR	R/W	1
14 (E)	CPU Identification	CPUID	R/W	2 Init
16 (10)	Process Control Block Base	PCBB	R/W	1
17 (11)	System Control Block Base	SCBB	R/W	1
18 (12)	Interrupt Priority Level	IPL	R/W	1 Init
19 (13)	AST Level	ASTLVL	R/W	1 Init
20 (14)	Software Interrupt Request	SIRR	WO	1

[1] See Table 4–1.

[2] Key to Classes:

1 = Implemented by the KA66A CPU module as specified in the *VAX Architecture Reference Manual*.
2 = Implemented uniquely by the KA66A CPU module.
3 = Accessible, but not fully implemented; accesses whem the system is in console mode are appropriate, accesses when the system is in user mode yield UNPREDICTABLE results.
n Init = The register is initialized on a KA66A CPU module reset (power-up, system reset, and node reset).

NOTE: Per-process registers, loaded by LDPCTX (load process context instruction), are the following IPRs (in decimal): 0, 1, 2, 3, 8, 9, 10, 11, 19, and 61. The remainder of the registers are not affected by LDPCTX.

Table 4–2 (Cont.): KA66A Internal Processor Registers

Address Dec. (Hex)	Register	Mnemonic	Type[1]	Class[2]
21 (15)	Software Interrupt Summary	SISR	R/W	1 Init
24 (18)	Interval Clock Control and Status[3]	ICCS	R/W	1 Init
25 (19)	Next Interval Count[3]	NICR	WO	2
26 (1A)	Interval Count[3]	ICR	RO	2
27 (1B)	Time-of-Day[4]	TODR	R/W	1
28 (1C)	Console Storage Receiver Status	CSRS	R/W	3 Init
29 (1D)	Console Storage Receiver Data	CSRD	RO	3 Init
30 (1E)	Console Storage Transmitter Status	CSTS	R/W	3 Init
31 (1F)	Console Storage Transmitter Data	CSTD	WO	3 Init
32 (20)	Console Receiver Control and Status	RXCS	R/W	2 Init
33 (21)	Console Receiver Data Buffer	RXDB	RO	2 Init
34 (22)	Console Transmitter Control and Status	TXCS	R/W	2 Init
35 (23)	Console Transmitter Data Buffer	TXDB	WO	2 Init
38 (26)	Machine Check Error Summary	MCESR	WO	2
42 (2A)	Console Saved Program Counter	SAVPC	RO	2

[1]See Table 4–1.

[2]Key to Classes:

1 = Implemented by the KA66A CPU module as specified in the *VAX Architecture Reference Manual*.
2 = Implemented uniquely by the KA66A CPU module.
3 = Accessible, but not fully implemented; accesses whem the system is in console mode are appropriate, accesses when the system is in user mode yield UNPREDICTABLE results.
n Init = The register is initialized on a KA66A CPU module reset (power-up, system reset, and node reset).

[3]Interval timer requests are posted at IPL 16 with a vector of C0 (hex). The interval timer is the lowest priority device at the IPL. A subset of ICCS is implemented in the NVAX chip. NICR and ICR can be used, depending on the settings in the Ebox Control Register.

[4]TODR is maintained during power failure by the XMI TOY BBU PWR line on the XMI backplane.

Table 4–2 (Cont.): KA66A Internal Processor Registers

Address Dec. (Hex)	Register	Mnemonic	Type[1]	Class[2]
43 (2B)	Console Saved Processor Status Longword	SAVPSL	RO	2
55 (37)	I/O Reset	IORESET	WO	2
56 (38)	Memory Management Enable	MAPEN	R/W	1 Init
57 (39)	Translation Buffer Invalidate All	TBIA	WO	1
58 (3A)	Translation Buffer Invalidate Single	TBIS	WO	1
62 (3E)	System Identification	SID	RO	2
63 (3F)	Translation Buffer Check	TBCHK	WO	1
64 (40)	IPL 14 Interrupt ACK	IAK14	RO	1
65 (41)	IPL 15 Interrupt ACK	IAK15	RO	1
66 (42)	IPL 16 Interrupt ACK	IAK16	RO	1
67 (43)	IPL 17 Interrupt ACK	IAK17	RO	1
68 (44)	Clear Write Buffer	CWB	R/W	1
122 (7A)	Interrupt System Status	INTSYS	R/W	2
124 (7C)	Patchable Control Store Control	PCSCR	R/W	2
125 (7D)	Ebox Control Register	ECR	R/W	2
160 (A0)	Cbox Control	CCTL	R/W	2 Init
162 (A2)	Backup Cache Data ECC	BCDECC	WO	2 Init
163 (A3)	Backup Cache Error Tag Status	BCETSTS	R/W	2
164 (A4)	Backup Cache Error Tag Index	BCETIDX	RO	2

[1]See Table 4–1.

[2]Key to Classes:

1 = Implemented by the KA66A CPU module as specified in the *VAX Architecture Reference Manual*.
2 = Implemented uniquely by the KA66A CPU module.
3 = Accessible, but not fully implemented; accesses whem the system is in console mode are appropriate, accesses when the system is in user mode yield UNPREDICTABLE results.
n Init = The register is initialized on a KA66A CPU module reset (power-up, system reset, and node reset).

Table 4–2 (Cont.): KA66A Internal Processor Registers

Address Dec. (Hex)	Register	Mnemonic	Type[1]	Class[2]
165 (A5)	Backup Cache Error Tag	BCETAG	RO	2
166 (A6)	Backup Cache Error Data Status	BCEDSTS	R/W	2
167 (A7)	Backup Cache Error Data Index	BCEDIDX	RO	2
168 (A8)	Backup Cache Error Data ECC	BCEDECC	RO	2
171 (AB)	Cbox Error Fill Address	CEFADR	RO	2
172 (AC)	Cbox Error Fill Status	CEFSTS	R/W	2
174 (AE)	NDAL Error Status	NESTS	R/W	2
176 (B0)	NDAL Error Output Address	NEOADR	RO	2
178 (B2)	NDAL Error Output Command	NEOCMD	RO	2
180 (B4)	NDAL Error Data High	NEDATHI	RO	2
182 (B6)	NDAL Error Data Low	NEDATLO	RO	2
184 (B8)	NDAL Error Input Command	NEICMD	RO	2
208 (D0)	VIC Memory Address	VMAR	R/W	2
209 (D1)	VIC Tag	VTAG	R/W	2
210 (D2)	VIC Data	VDATA	R/W	2
211 (D3)	Ibox Control and Status	ICSR	R/W	2
212 (D4)	Ibox Branch Prediction Control	BPCR	R/W	2
214 (D6)	Ibox Backup PC	BPC	RO	2
215 (D7)	Ibox Backup PC with RLOG Unwind	BPCUNW	RO	2

[1]See Table 4–1.

[2]Key to Classes:

1 = Implemented by the KA66A CPU module as specified in the *VAX Architecture Reference Manual*.
2 = Implemented uniquely by the KA66A CPU module.
3 = Accessible, but not fully implemented; accesses whem the system is in console mode are appropriate, accesses when the system is in user mode yield UNPREDICTABLE results.
n Init = The register is initialized on a KA66A CPU module reset (power-up, system reset, and node reset).

Table 4–2 (Cont.): KA66A Internal Processor Registers

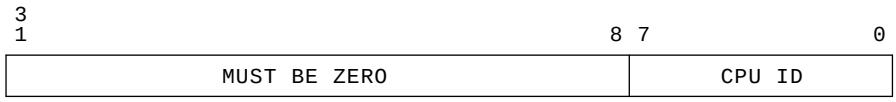
Address Dec. (Hex)	Register	Mnemonic	Type[1]	Class[2]
231 (E7)	Physical Address Mode	PAMODE	R/W	2
232 (E8)	Memory Management Exception Address	MMEADR	RO	2
233 (E9)	Memory Management Exception PTE Address	MMEPTE	RO	2
234 (EA)	Memory Management Exception Status	MMESTS	RO	2
236 (EC)	TB Parity Address	TBADR	RO	2
237 (ED)	TB Parity Status	TBSTS	R/W	2
242 (F2)	P-Cache Parity Address	PCADR	RO	2
244 (F4)	P-Cache Status	PCSTS	R/W	2
248 (F8)	P-Cache Control	PCCTL	R/W	2

[1] See Table 4–1.

[2] Key to Classes:

1 = Implemented by the KA66A CPU module as specified in the *VAX Architecture Reference Manual*.
2 = Implemented uniquely by the KA66A CPU module.
3 = Accessible, but not fully implemented; accesses whem the system is in console mode are appropriate, accesses when the system is in user mode yield UNPREDICTABLE results.
n Init = The register is initialized on a KA66A CPU module reset (power-up, system reset, and node reset).

Figure 4–1: CPU Identification Register (CPUID) IPR14 (E)



msb–p504–91

Figure 4–2: Interval Clock Control and Status Register (ICCS) IPR24 (18)

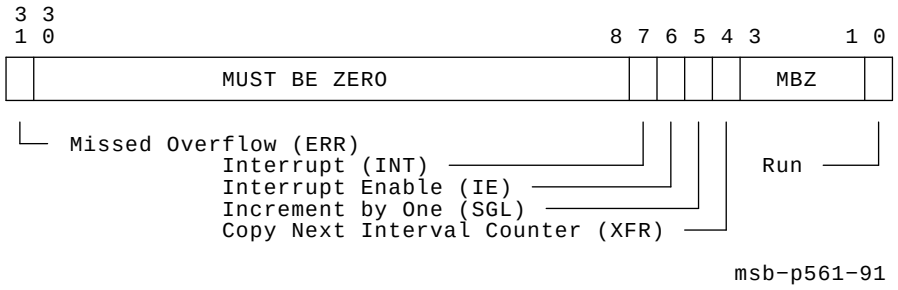


Figure 4–3: Next Interval Count Register (NICR) IPR25 (19)

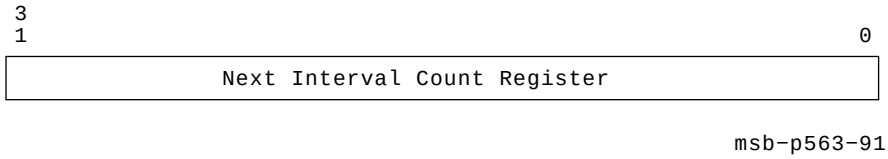


Figure 4–4: Interval Count Register (ICR) IPR26 (1A)

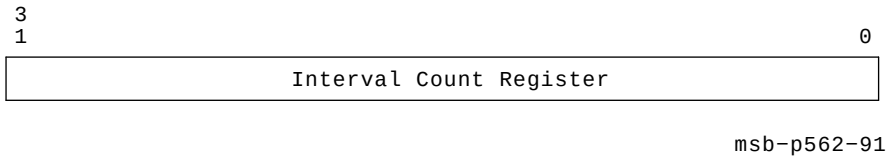


Figure 4–5: Console Receiver Control and Status Register (RXCS) IPR32 (20)

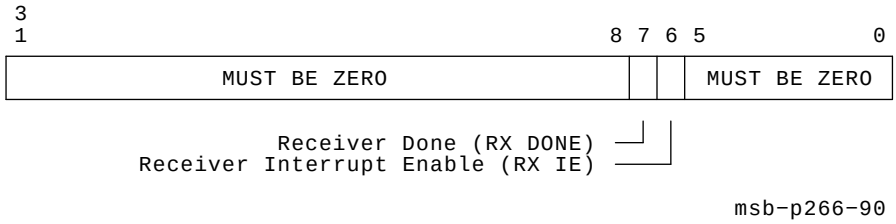


Figure 4–6: Console Receiver Data Buffer Register (RXDB) IPR33 (21)

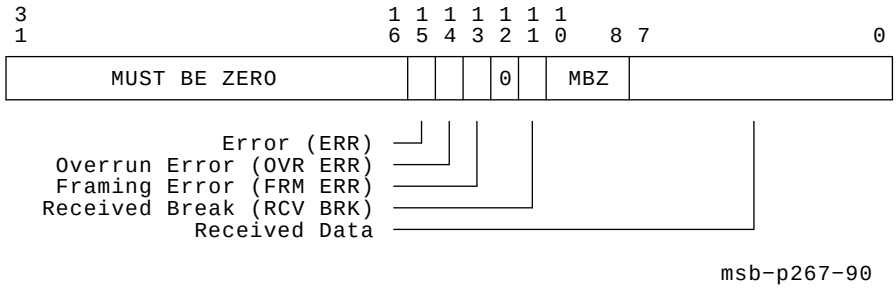


Figure 4–7: Console Transmitter Control and Status Register (TXCS) IPR34 (22)

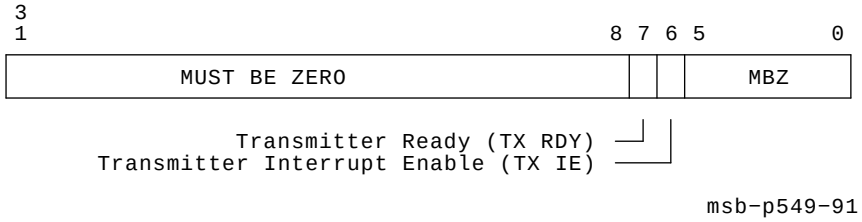


Figure 4–8: Console Transmitter Data Buffer Register (TXDB) IPR35 (23)

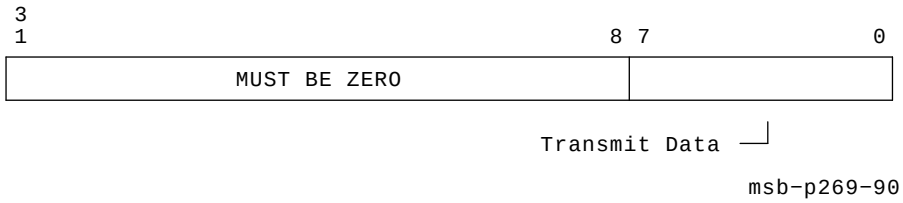


Figure 4–9: Machine Check Error Summary Register (MCESR) IPR38 (26)

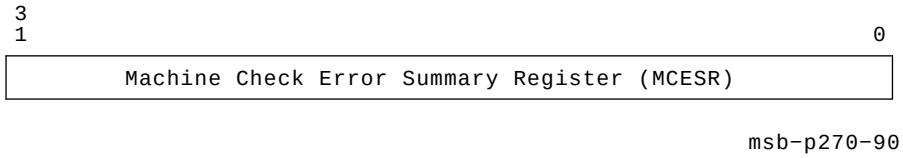


Figure 4–10: Console Saved Program Counter Register (SAVPC) IPR42 (2A)

Figure 4–11: Console Saved Processor Status Longword (SAVPSL) IPR43 (2B)

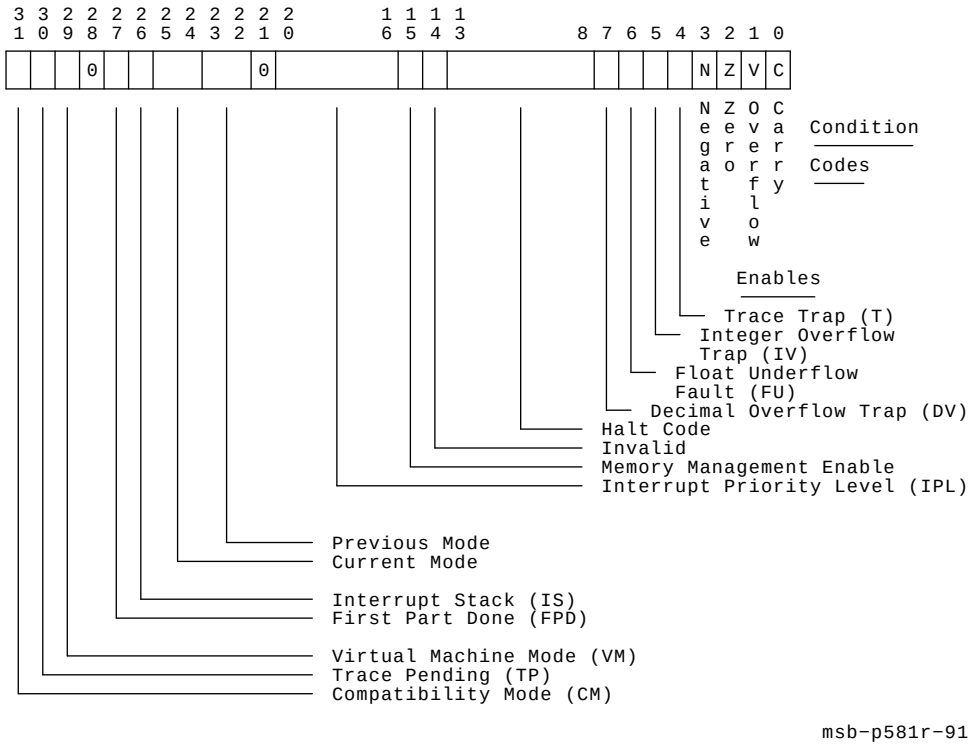
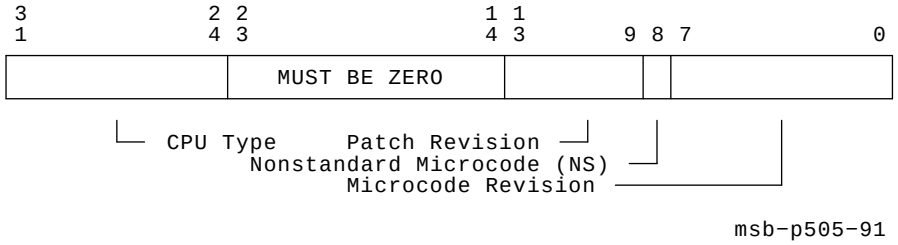


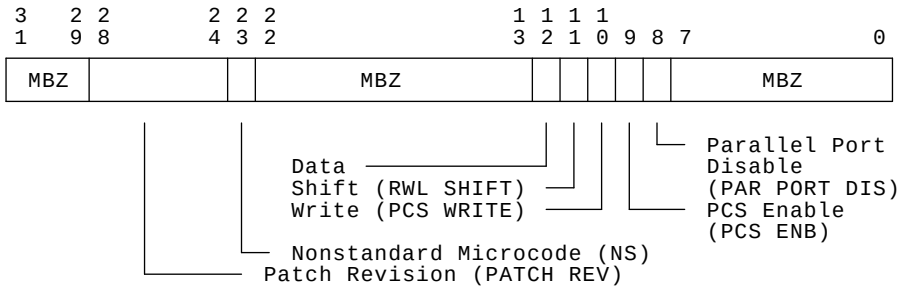
Figure 4–12: I/O Reset Register (IORESET) IPR55 (37)

Figure 4–13: System Identification Register (SID) IPR62 (3E)



msb–p505–91

Figure 4–14: Patchable Control Store Control Register (PCSCR) IPR124 (7C)



msb–p513–91

Figure 4–15: Ebox Control Register (ECR)
IPR125 (7D)

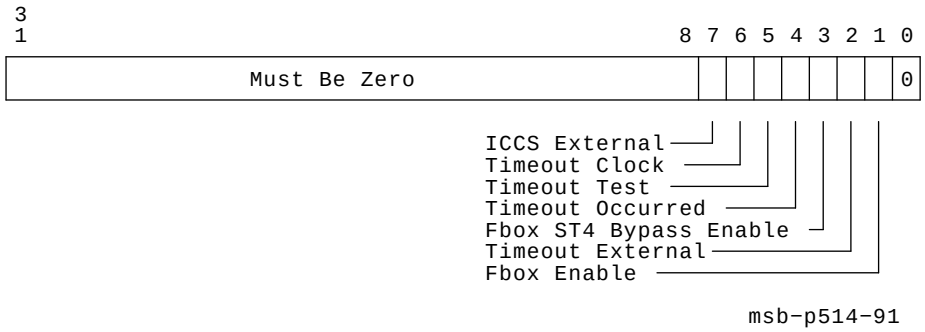


Figure 4–16: Cbox Control Register (CCTL)
IPR160 (A0)

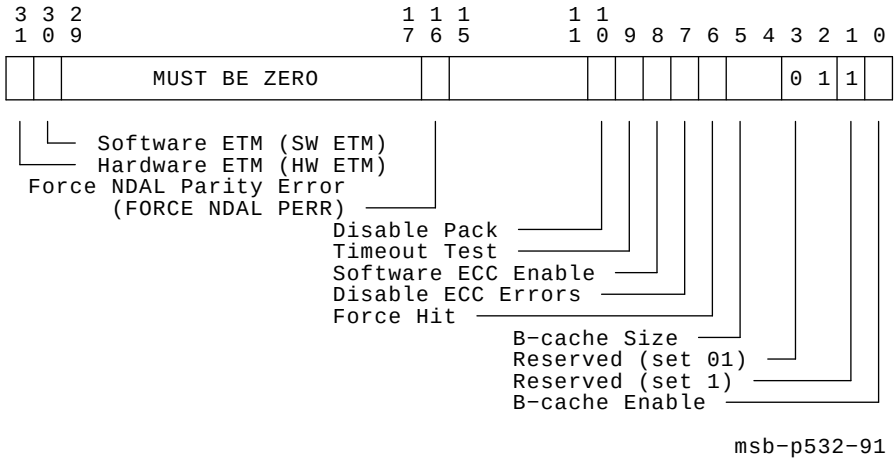


Figure 4–17: Backup Cache Data ECC Register (BCDECC) IPR162 (A2)

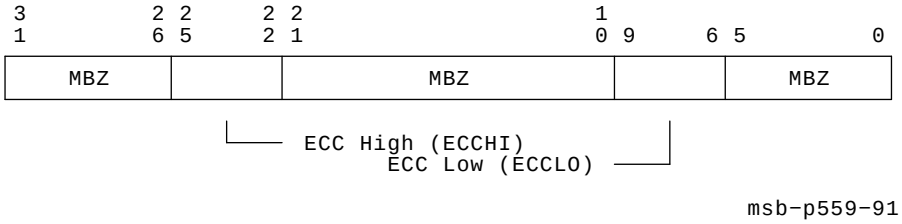


Figure 4–18: Backup Cache Error Tag Status Register (BCETSTS) IPR163 (A3)

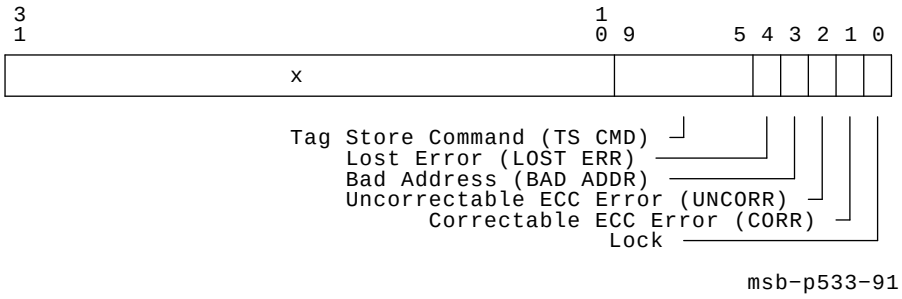


Figure 4–19: Backup Cache Error Tag Index Register (BCETIDX) IPR164 (A4)

Figure 4–20: Backup Cache Error Tag Register (BCETAG) IPR165 (A5)

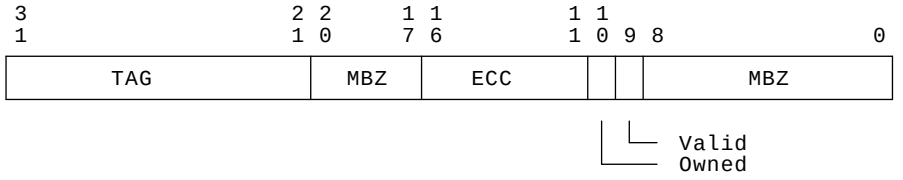


Figure 4–21: Backup Cache Error Data Status Register (BCEDSTS) IPR166 (A6)

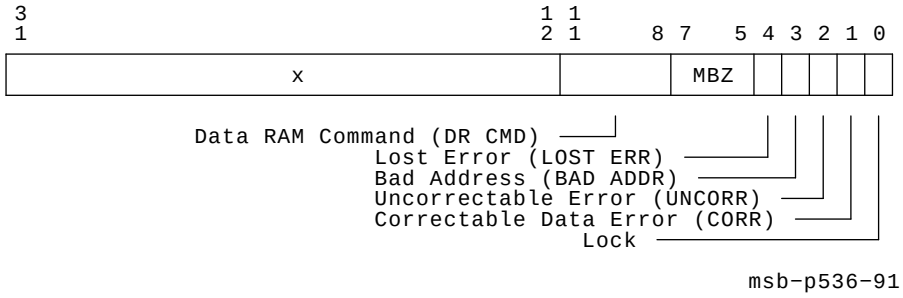


Figure 4–22: Backup Cache Error Data Index Register (BCEDIDX) IPR167 (A7)

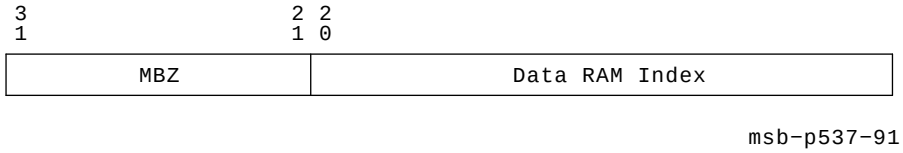


Figure 4–23: Backup Cache Error Data ECC Register (BCEDECC) IPR168 (A8)

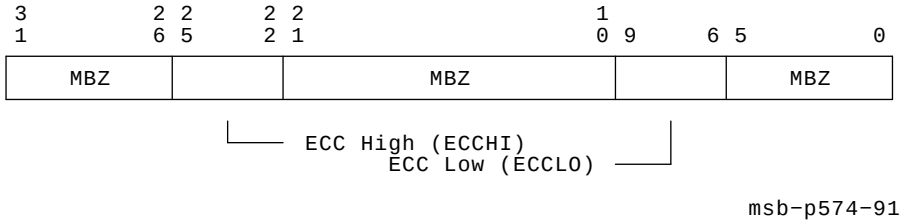


Figure 4–24: Cbox Error Fill Address Register (CEFADR) IPR171 (AB)

Figure 4–25: Cbox Error Fill Status Register (CEFSTS) IPR172 (AC)

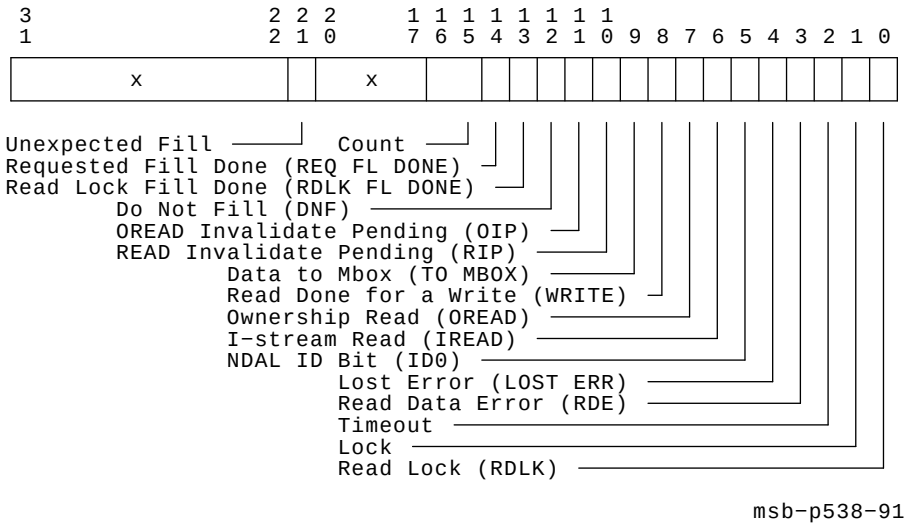


Figure 4–26: NDAL Error Status Register (NESTS) IPR174 (AE)

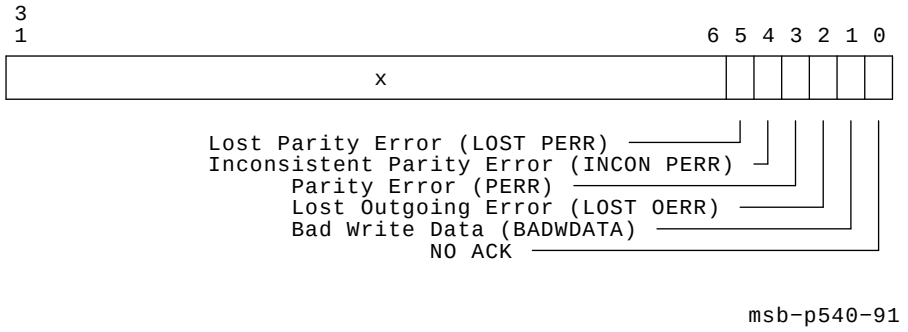
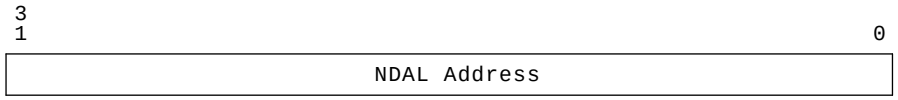
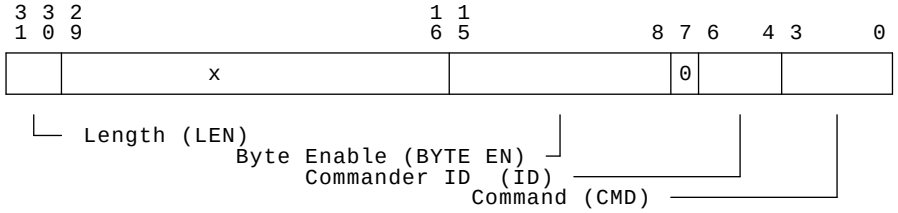


Figure 4–27: NDAL Error Output Address Register (NEOADR) IPR176 (B0)



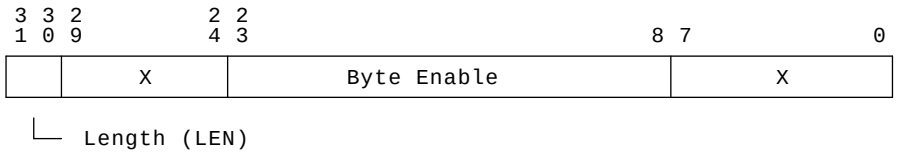
msb–p541–91

Figure 4–28: NDAL Error Output Command Register (NEOCMD) IPR178 (B2)



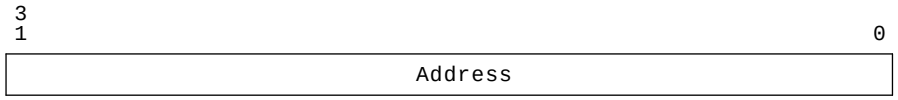
msb–p542–91

Figure 4–29: NDAL Error Data High Register (NEDATHI) IPR180 (B4)



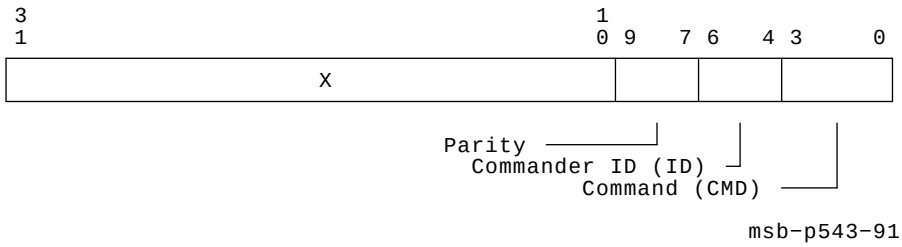
msb–p544–91

Figure 4–30: NDAL Error Data Low Register (NEDATLO) IPR182 (B6)



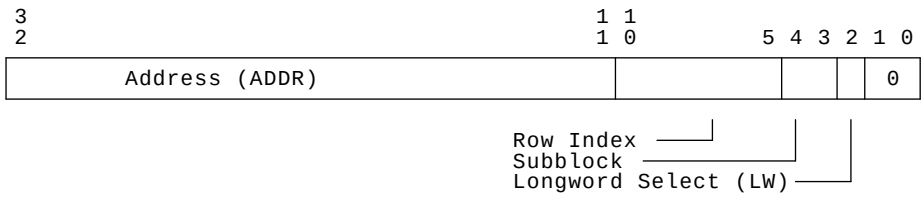
msb–p545–91

Figure 4–31: NDAL Error Input Command Register (NEICMD) IPR184 (B8)



msb–p543–91

Figure 4–32: VIC Memory Address Register (VMAR) IPR208 (D0)



msb–p509–91

Figure 4–33: VIC Tag Register (VTAG) IPR209 (D1)

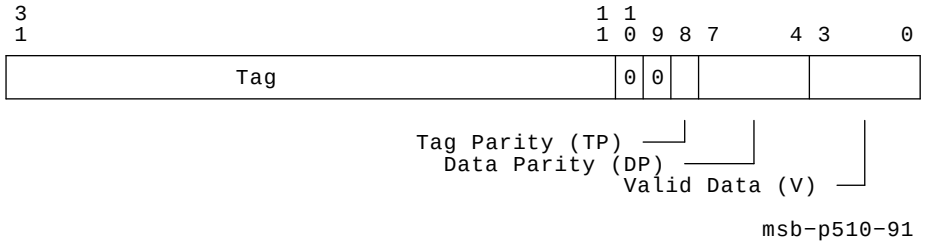


Figure 4–34: VIC Data Register (VDATA) IPR210 (D2)

Figure 4–35: Ibox Control and Status Register (ICSR) IPR211 (D3)

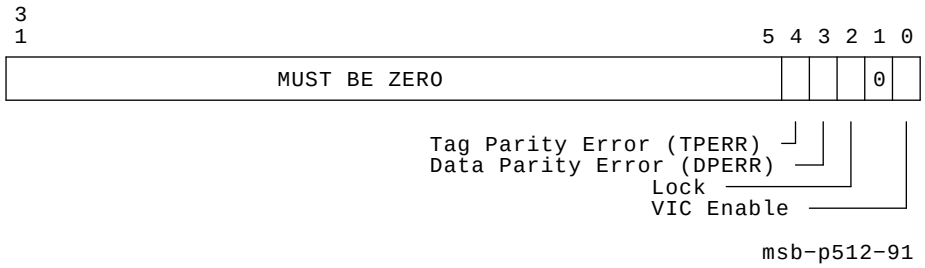


Figure 4–36: Physical Address Mode Register (PAMODE) IPR231 (E7)

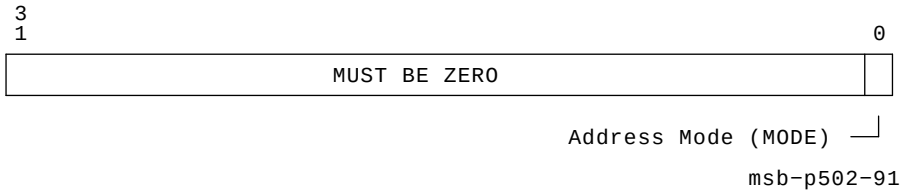


Figure 4–37: Memory Management Exception Address Register (MMEADR) IPR232 (E8)

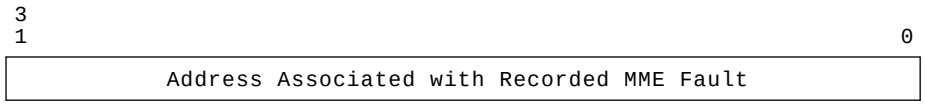


Figure 4–38: Memory Management Exception PTE Address Register (MMEPTE) IPR233 (E9)

Figure 4–39: Memory Management Exception Status Register (MMESTS) IPR234 (EA)

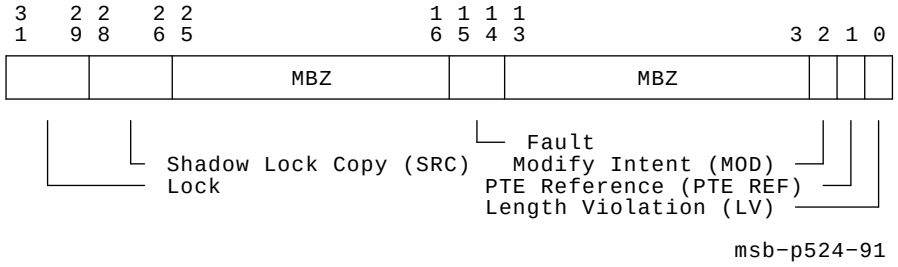


Figure 4–40: TB Parity Address Register (TBADR) IPR236 (EC)

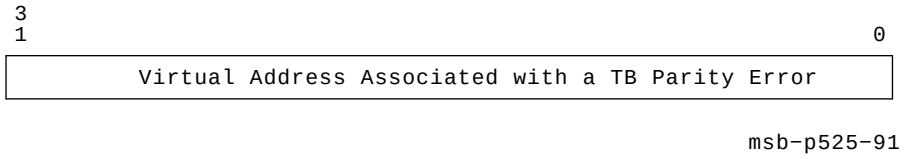


Figure 4–41: TB Parity Status Register (TBSTS) IPR237 (ED)

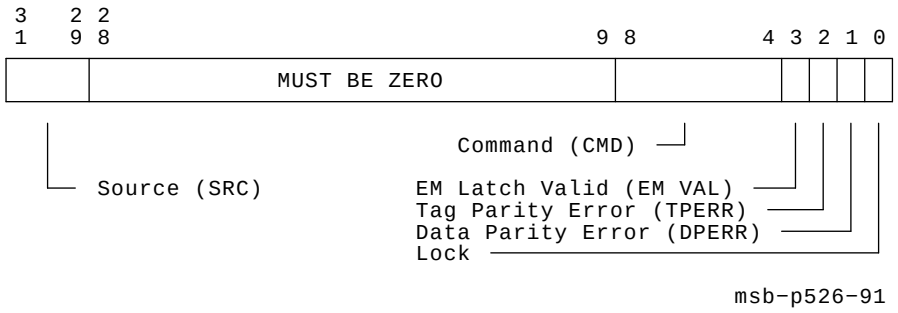
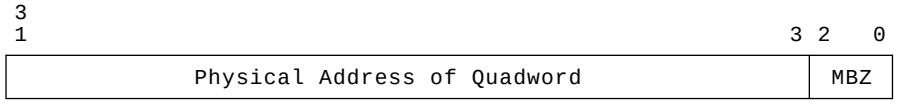
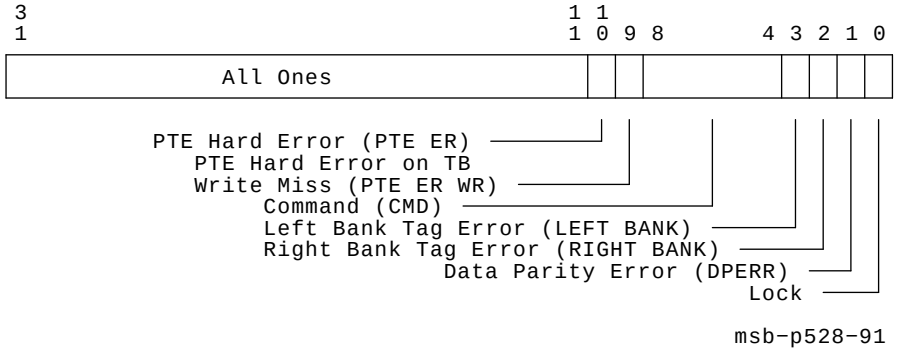


Figure 4–42: P-Cache Parity Address Register (PCADR) IPR242 (F2)



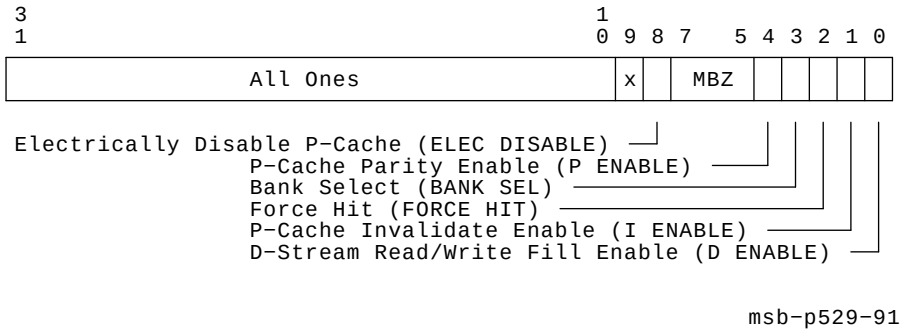
msb–p527–91

Figure 4–43: P-Cache Status Register (PCSTS) IPR244 (F4)



msb–p528–91

Figure 4–44: P-Cache Control Register (PCCTL) IPR248 (F8)



msb–p529–91

4.2 KA66A Registers in XMI Private Space

Table 4–3: KA66A Registers in XMI Private Space

Register	Mnemonic	Address
NDAL CSR	NCSR	E000 0000
TOY Clock Registers		E018 3000 – E018 300D
BBU RAM		E018 300E – E018 303F
NEXMI Input Port	IPORT	E018 4000
NEXMI Output Port0	OPORT0	E018 5000
NEXMI Output Port1	OPORT1	E018 6000
UART Registers		E018 7000 – E018 700F
IPR Address Space		E100 0000 – E100 03FF
IP IVINTR Generation	IPINTR	E101 0000 – E101 FFFF
WE IVINTR Generation	WEINTR	E102 0000 – E102 FFFF

Figure 4–45: NDAL Control and Status Register (NCSR) E000 0000

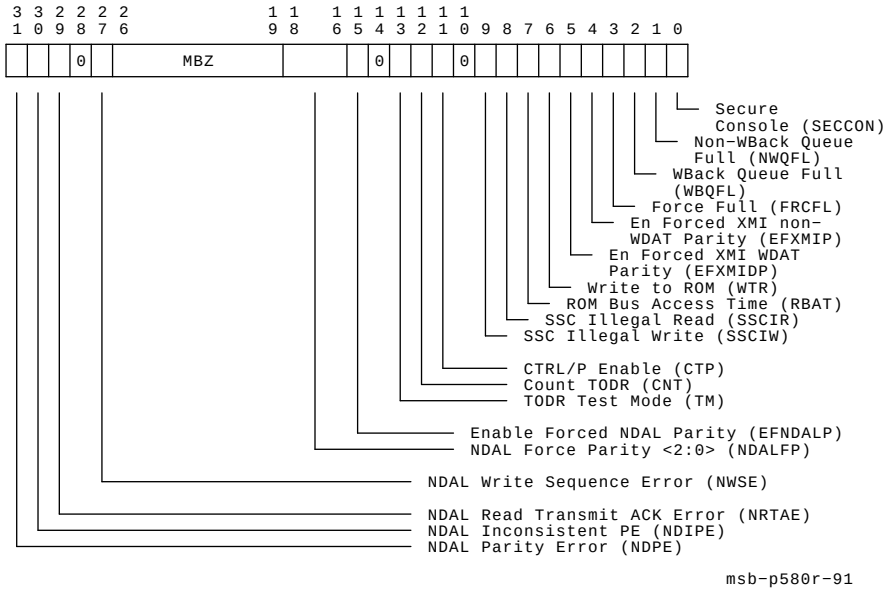


Figure 4–46: NEXMI Input Port Register (IPORT) E018 4000

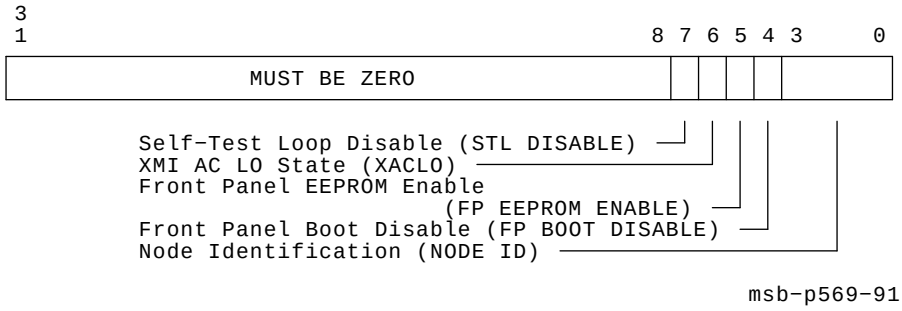


Figure 4–47: NEXMI Output Port0 Register (OPORT0) E018 5000

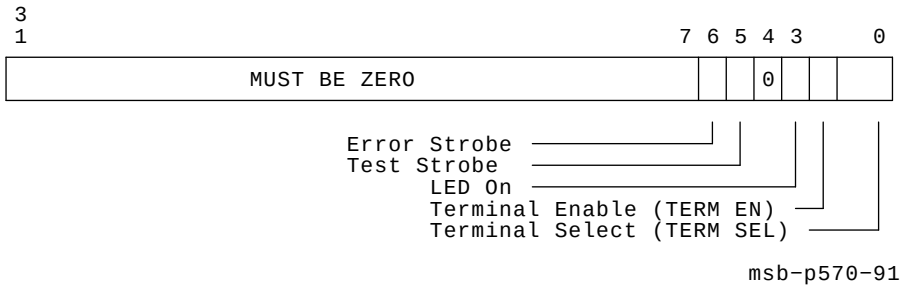
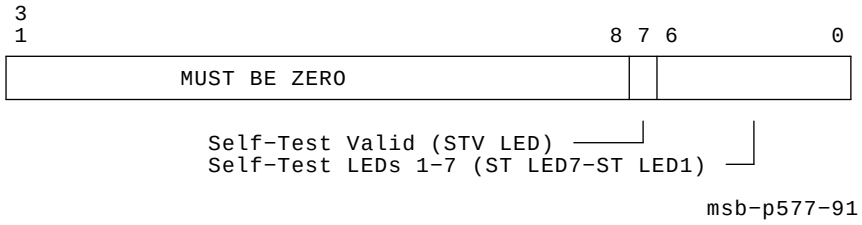


Figure 4–48: NEXMI Output Port1 Register (OPORT1) E018 6000



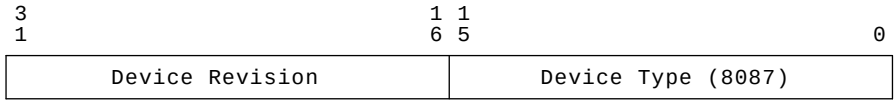
4.3 KA66A XMI Registers

Table 4–4: XMI Registers for the KA66A CPU Module

Register	Mnemonic	Address
Device Register	XDEV	BB[1] + 00
Bus Error	XBER	BB + 04
Failing Address	XFADR	BB + 08
XMI General Purpose	XGPR	BB + 0C
Node-Specific Control and Status	NSCSR	BB + 1C
XMI Control Register	XCR	BB + 24
Failing Address Extension	XFAER	BB + 2C
Bus Error Extension	XBEER	BB + 34
Writeback 0 Failing Address	WFADR0	BB + 40
Writeback 1 Failing Address	WFADR1	BB + 44

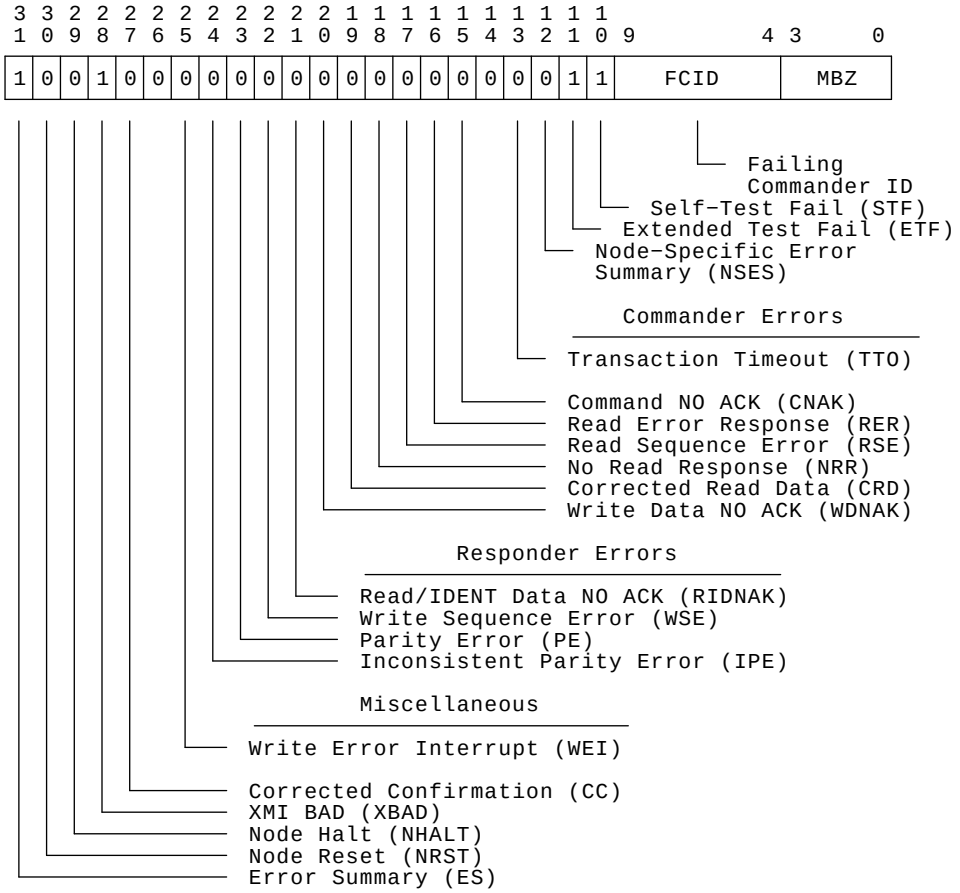
[1]BB = base address of a node, which is the address of the first location in nodespace.

Figure 4–49: Device Register (XDEV) BB + 00



msb-p553-91

Figure 4–50: Bus Error Register (XBER)
BB + 04

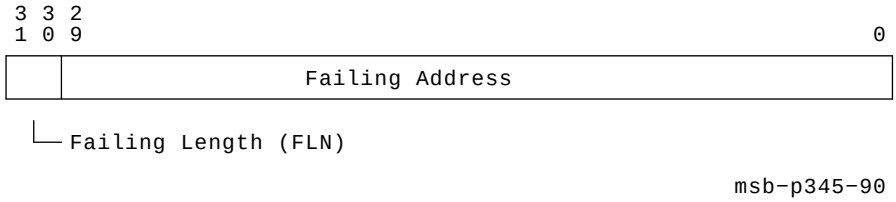


The bit values shown are initialized settings.

msbp-579-91

XFADR, when the XMI command is neither an IDENT transaction nor an IVINTR transaction:

Figure 4–51: Failing Address Register (XFADR)
BB + 08

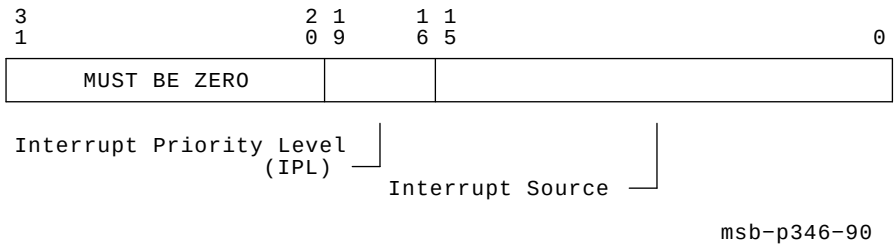


NOTE: *When XFADR contains a read or write address, the bit map for a 32-bit DAL address is as follows:*

If XFADR<29> = 0 (memory space) then 32-bit DAL address is: XFADR<29> + XFAER<17:16> + XFADR<28:0>.

If XFADR<29> = 1 (I/O space) then 32-bit DAL address is: 111 + XFADR<28:0>.

XFADR, when the XMI command is 9 (hex), an IDENT transaction:



XFADR, when the XMI command is F (hex), an IVINTR transaction:

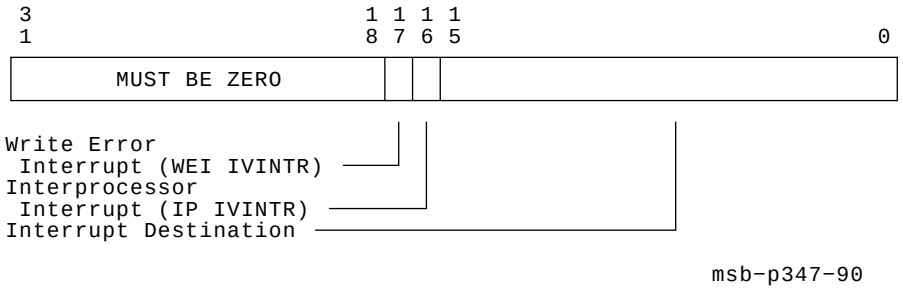


Figure 4–52: XMI General Purpose Register (XGPR)
BB + 0C

Figure 4–53: Node-Specific Control and Status Register (NSCSR)
BB + 1C

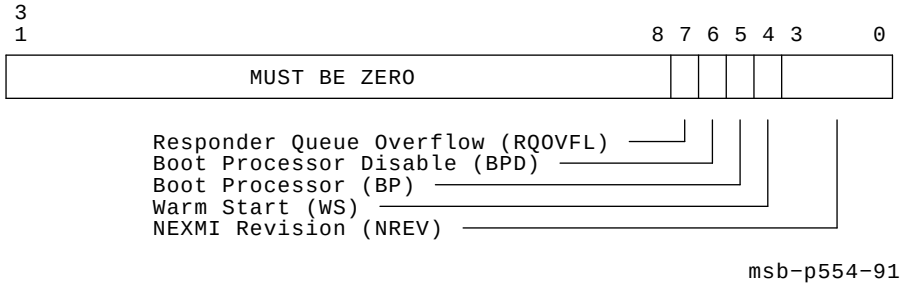
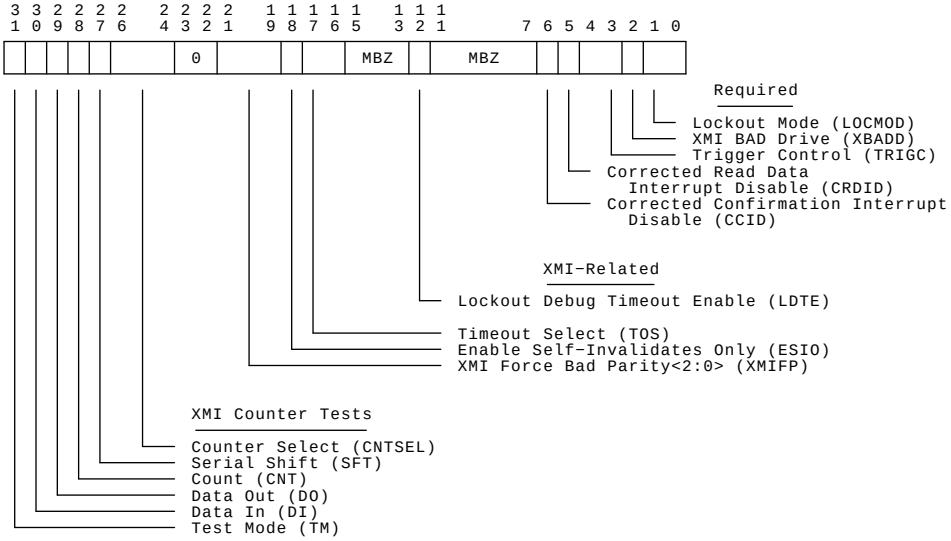
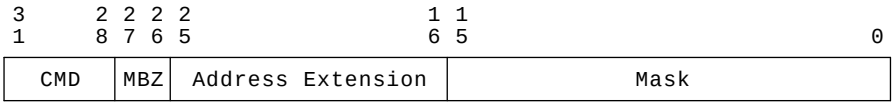


Figure 4–54: XMI Control Register (XCR)
BB + 24



msb–p555–91

Figure 4–55: Failing Address Extension Register (XFAER)
BB + 2C



msb–p556–91

Figure 4–56: Bus Error Extension Register (XBEER)
BB + 34

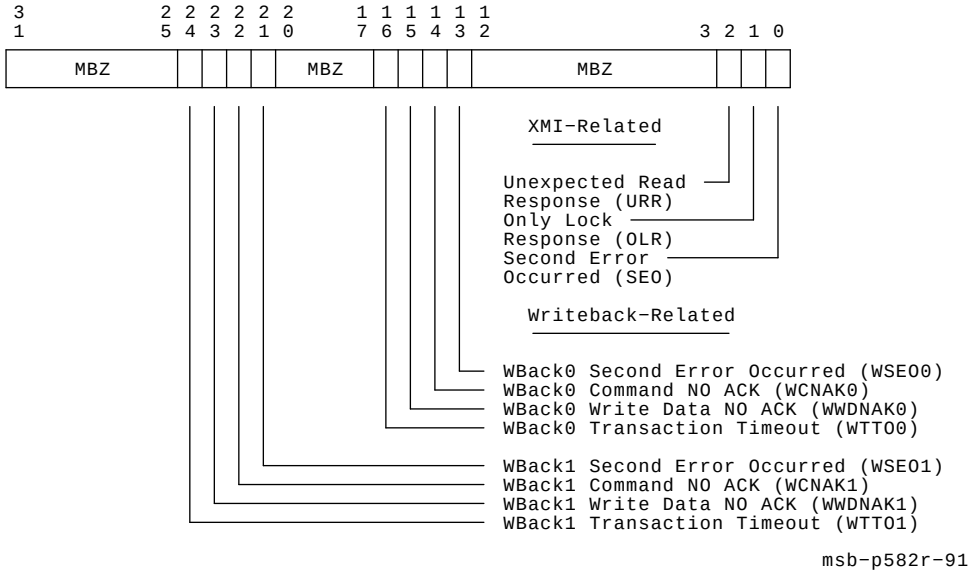


Figure 4–57: Writeback 0 Failing Address Register (WFADR0)
BB + 40

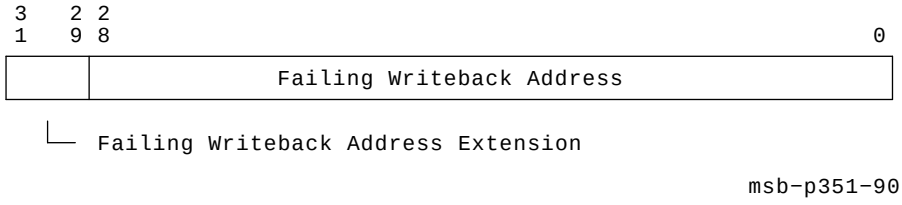
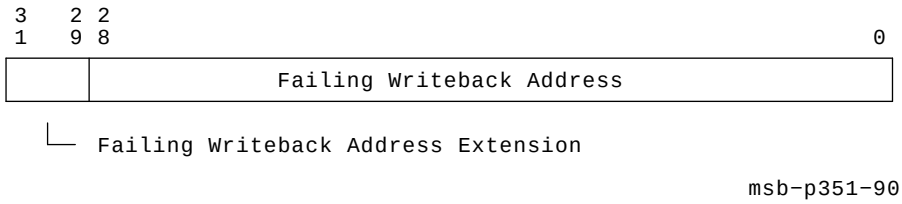


Figure 4–58: Writeback 1 Failing Address Register (WFADR1) BB + 44

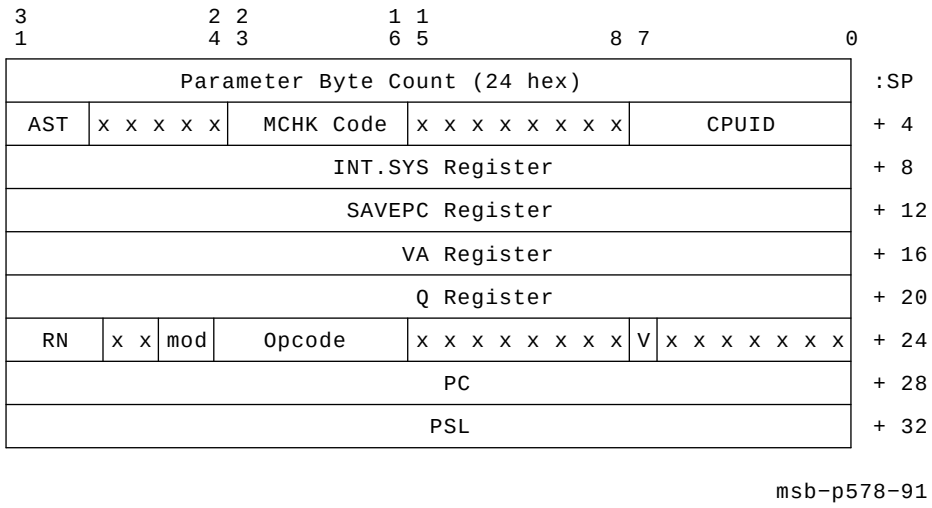


4.4 Machine Checks

A machine check exception is reported through SCB vector 04 (hex) when the NVAX chip detects an error condition. The frame pushed on the stack for a machine check indicates the type of error and provides internal state information that may help identify the cause of the error. The machine check stack frame is shown in Figure 4–59 and its parameters are described in Table 4–5. Table 4–6 lists and describes the machine check codes.

Software must acknowledge machine checks by writing a zero to IPR38, MCESR.

Figure 4–59: Machine Check Stack Frame



msb–p578–91

Table 4–5: Machine Check Stack Frame Fields

Longword	Bits	Contents
SP+0	<31:0>	Byte count— The size of the stack frame in bytes, not including the PC, PSL, or the byte count longword. Stack frame PC and PSL values should always be referenced using this count as an offset from the stack pointer.
SP+4	<31:29>	ASTLVL— The current value of the register.
	<23:16>	Machine check code— The reason for the machine check, as listed in Table 4–6.
	<7:0>	CPUID—The current value of the CPUID register.
SP+8	<31:0>	INT.SYS register— The value of the INT.SYS register and read onto the A-bus by the microcode.
SP+12	<31:0>	SAVEPC— The SAVEPC register which is loaded by microcode with the PC value in certain circumstances. It is used in error handling for PTE read errors with PSL<FPD> set in this stack frame.
SP+16	<31:0>	VA register— The contents of the Ebox VA register, which may be loaded from the output of the ALU.
SP+20	<31:0>	Q register— The contents of the Ebox Q register, which may be loaded from the output of the shifter.
SP+24	<31:28>	Rn— The value of the Rn register, which is used to obtain the register number for the CVTPL and EDIV instructions. In general, the value of this field is UNPREDICTABLE.
	<25:24>	Mode— A copy of PSL<CUR MOD>.
	<23:16>	Opcode— Bits <7:0> of the instruction opcode. The FD bit is not included.
	<7>	VR— The VAX Restart bit, which is used to communicate restart information between the microcode and the operating system. If this bit is set, no architectural state has been changed by the instruction which was executing when the error was detected. If this bit is not set, architectural state was modified by the instruction.
SP+28	<31:0>	PC— The value of the program counter at the time of the fault.
SP+32	<31:0>	PSL— The value of the processor status longword at the time of the fault.

Table 4–6: Machine Check Codes

Code (hex)	Mnemonic	Description
01	MCHK_UNKNOWN_MSTATUS	Unknown memory management fault parameter returned by Mbox
02	MCHK_INIT.ID_VALUE	Illegal interrupt ID value returned in INT.SYS
03	MCHK_CANT_GET_HERE	Illegal microcode dispatch occurred
04	MCHK_MOVC.STATUS	Illegal combination of state bits detected during string instruction
05	MCHK_ASYNC_ERROR	Asynchronous hardware error occurred
06	MCHK_SYNC_ERROR	Synchronous hardware error occurred

4.5 KA66A Parse Trees

Figure 4–60: Machine Check Parse Tree

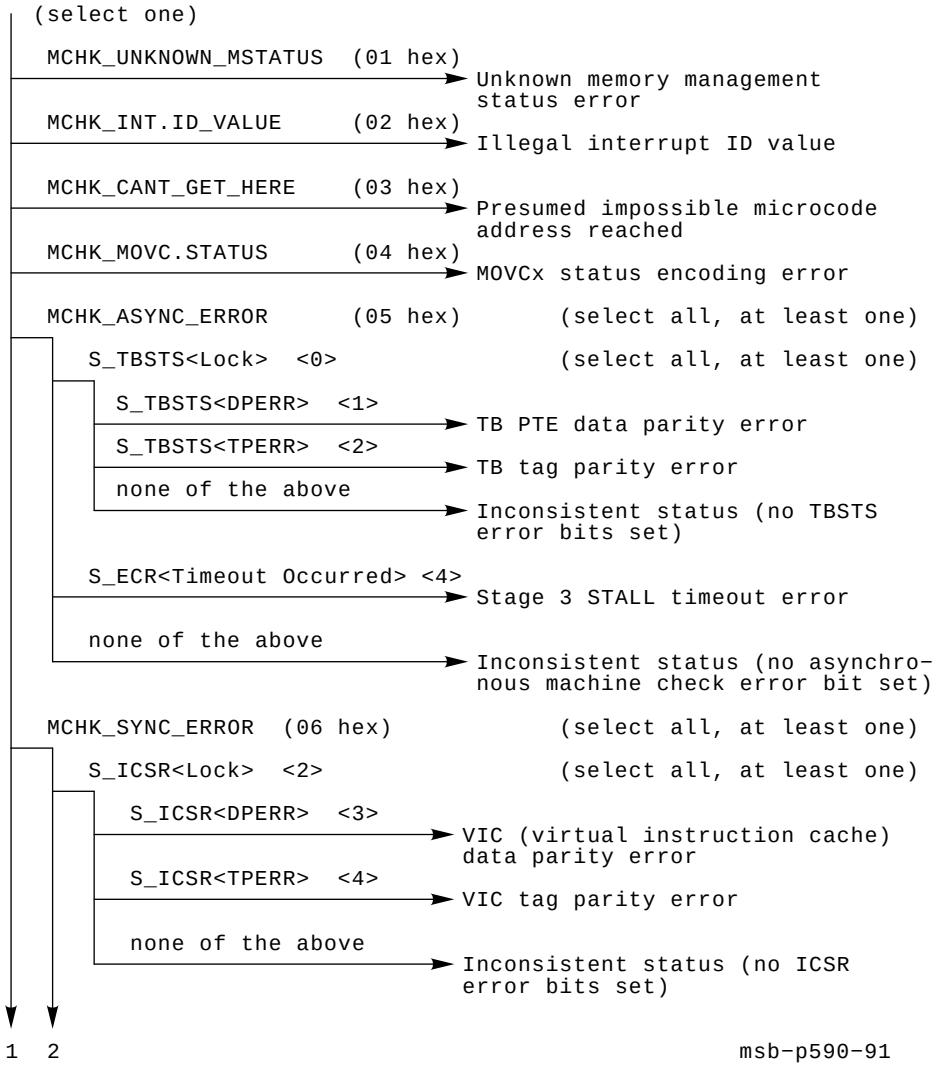


Figure 4–60 Cont'd on next page

Figure 4–60 (Cont.): Machine Check Parse Tree

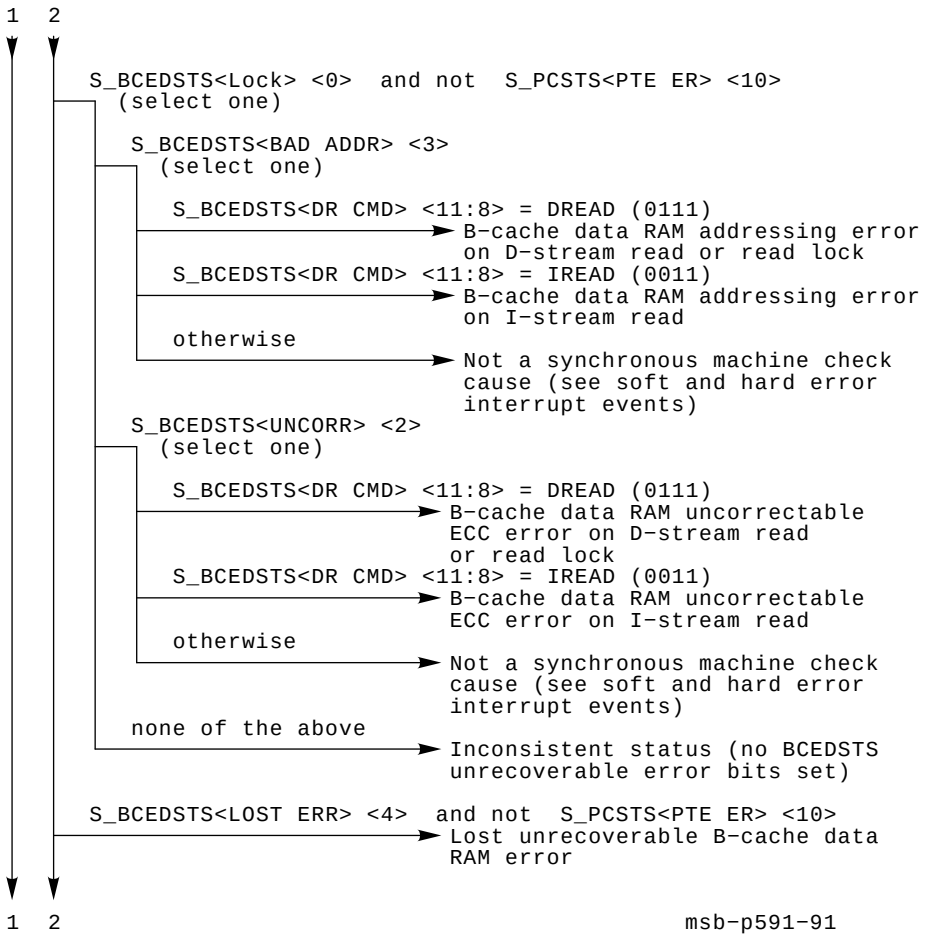


Figure 4–60 Cont'd on next page

Figure 4–60 (Cont.): Machine Check Parse Tree

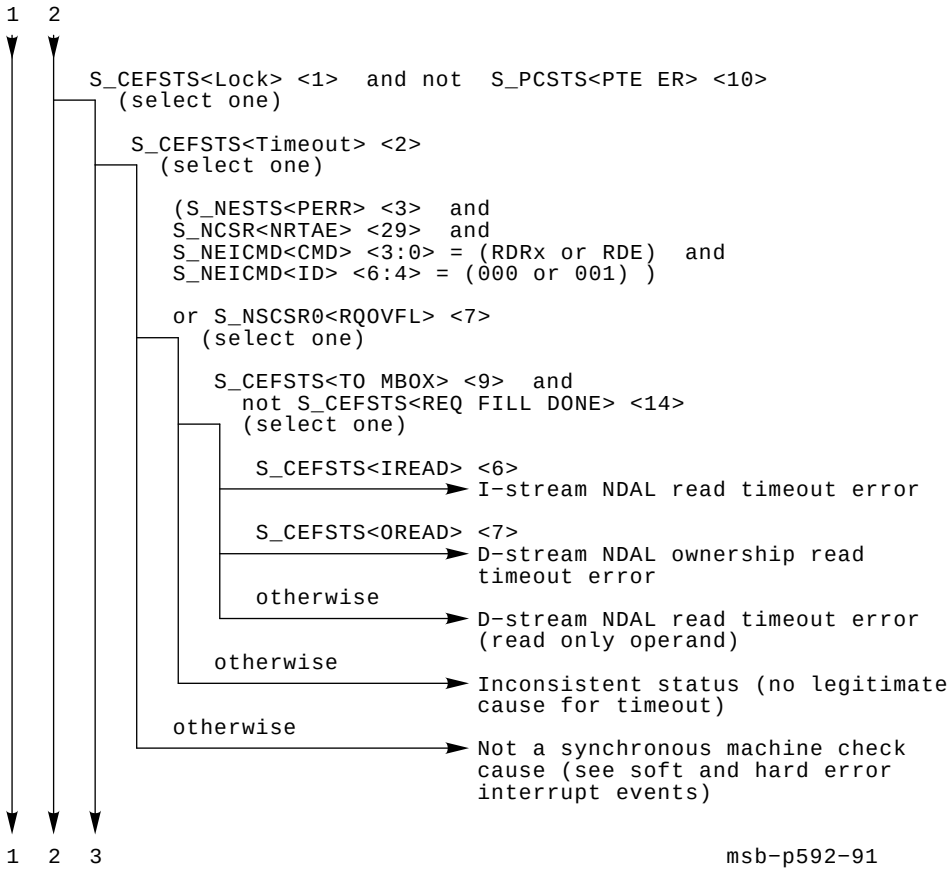


Figure 4–60 Cont'd on next page

Figure 4–60 (Cont.): Machine Check Parse Tree

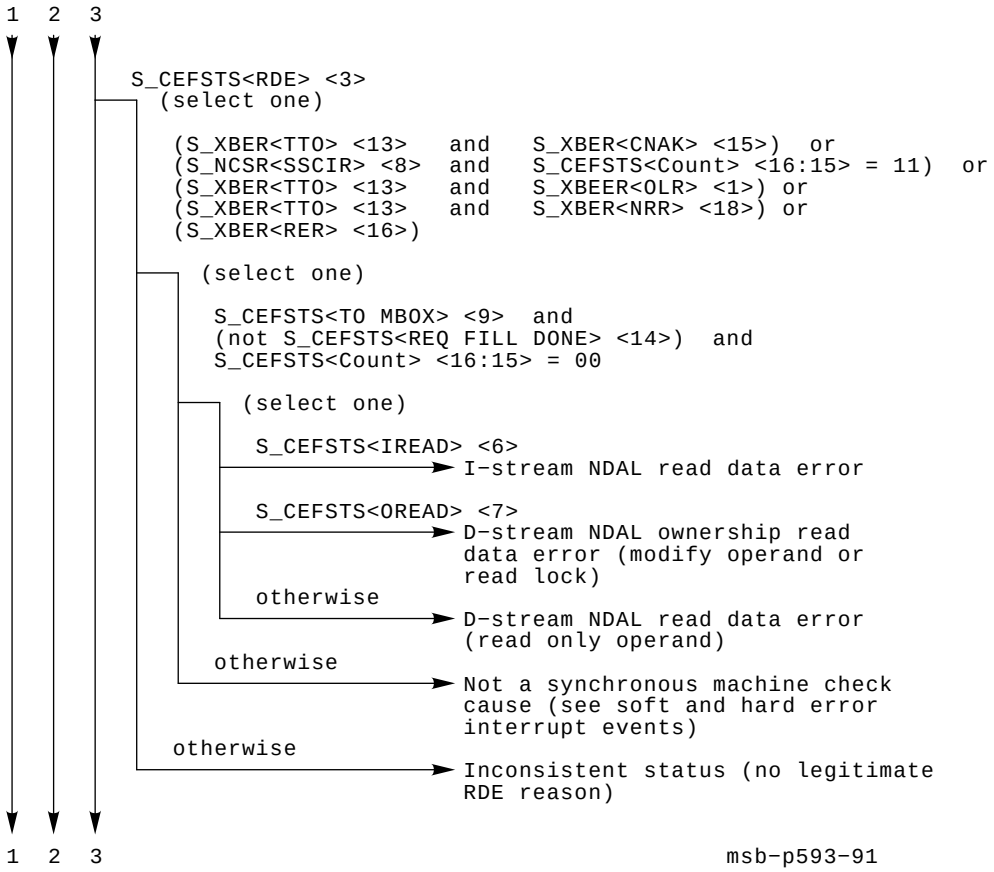


Figure 4–60 Cont'd on next page

Figure 4–60 (Cont.): Machine Check Parse Tree

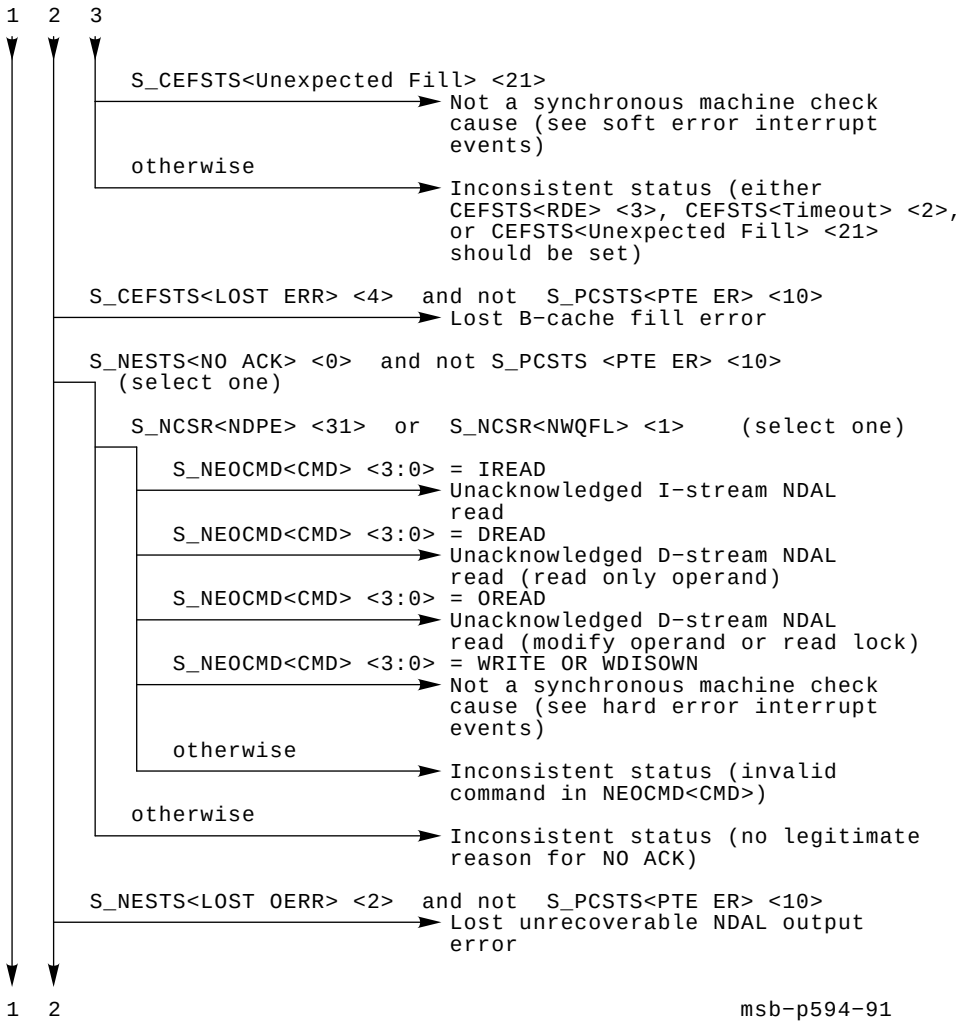


Figure 4–60 Cont’d on next page

Figure 4–60 (Cont.): Machine Check Parse Tree

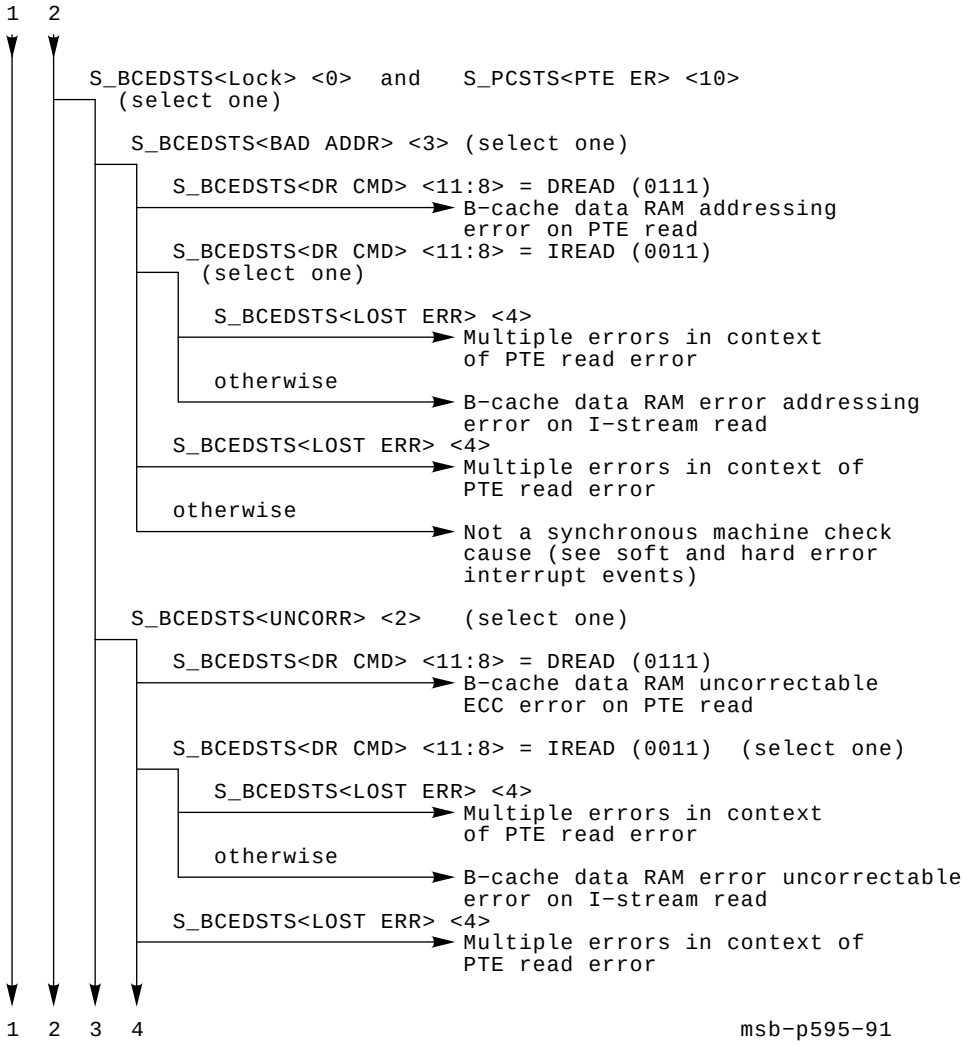


Figure 4–60 Cont'd on next page

Figure 4–60 (Cont.): Machine Check Parse Tree

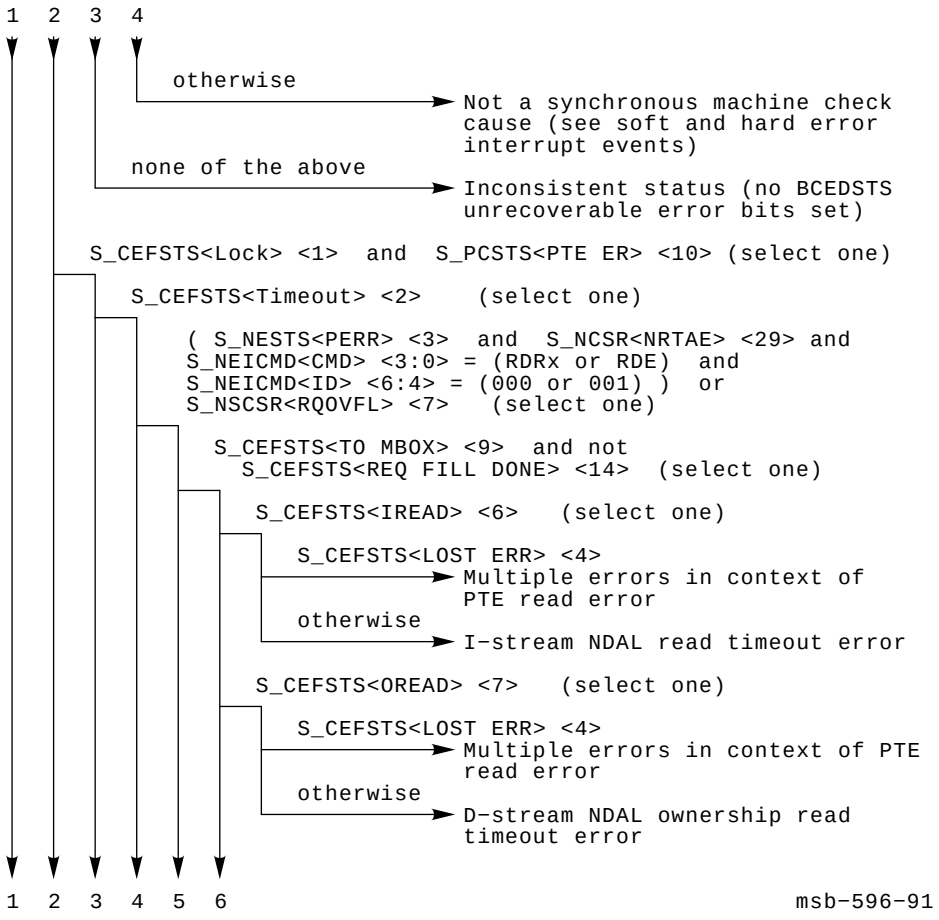


Figure 4–60 Cont'd on next page

Figure 4–60 (Cont.): Machine Check Parse Tree

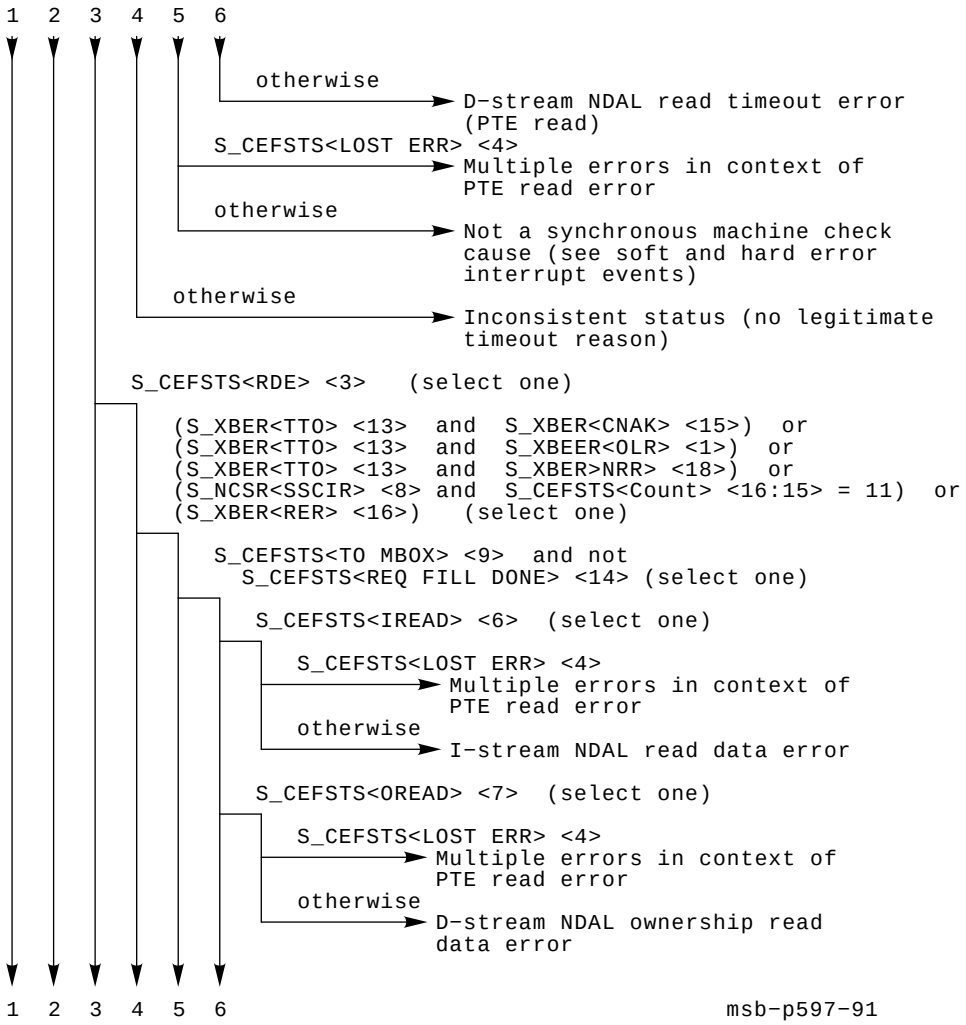


Figure 4–60 Cont'd on next page

Figure 4–60 (Cont.): Machine Check Parse Tree

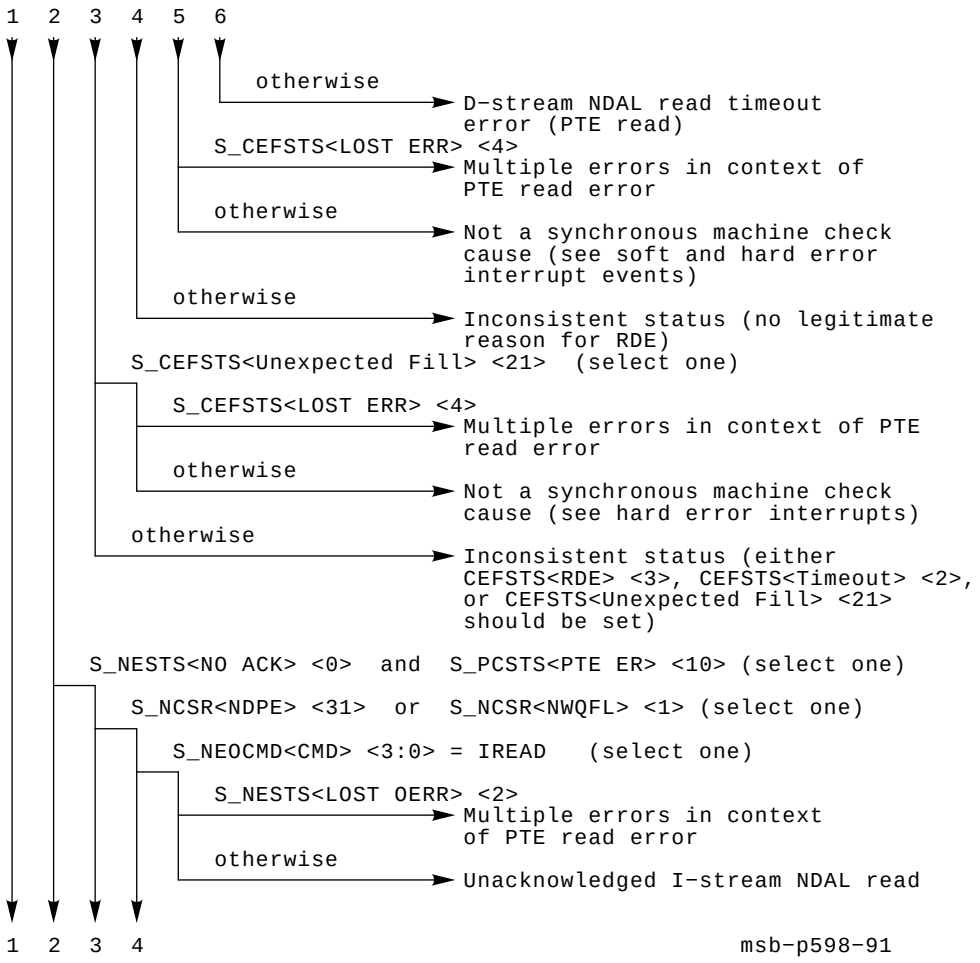


Figure 4–60 Cont'd on next page

Figure 4–60 (Cont.): Machine Check Parse Tree

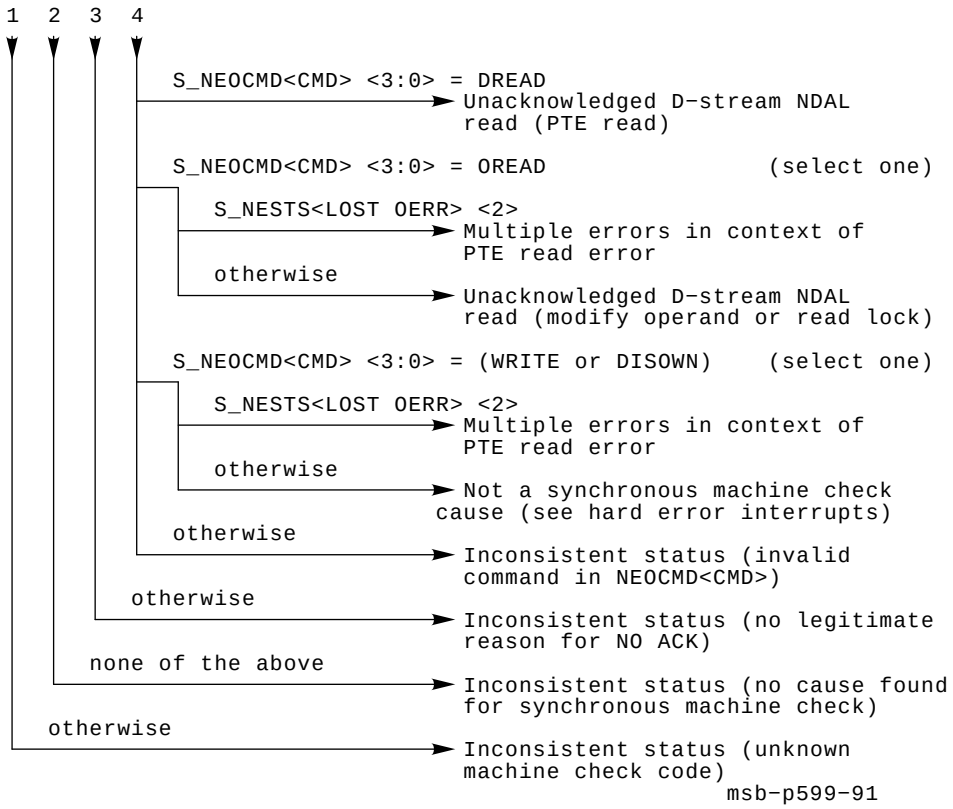


Figure 4–61: Hard Error Interrupt Parse Tree

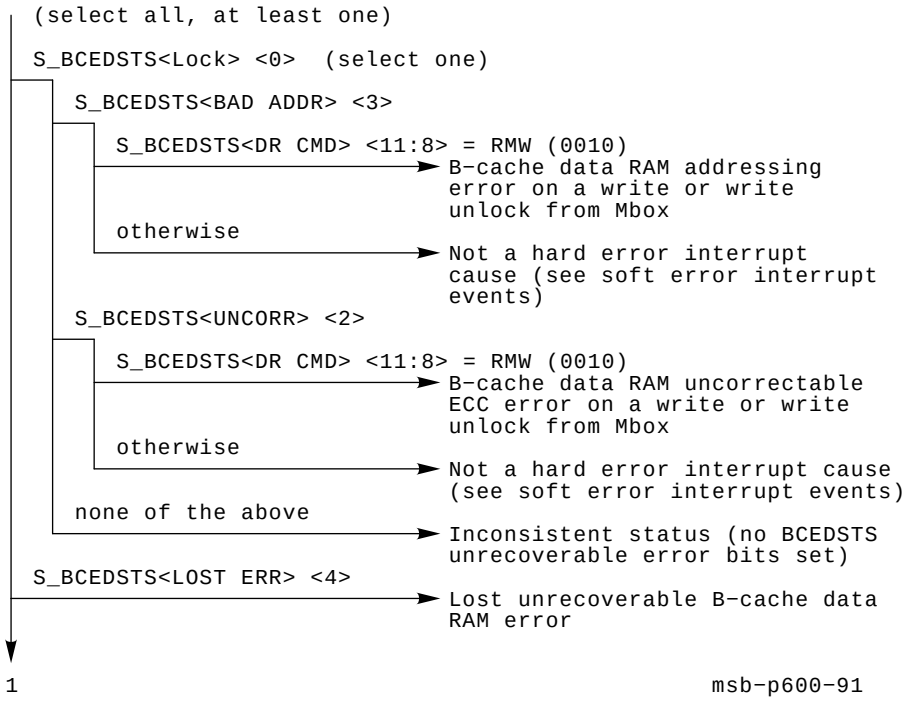


Figure 4–61 Cont'd on next page

Figure 4–61 (Cont.): Hard Error Interrupt Parse Tree

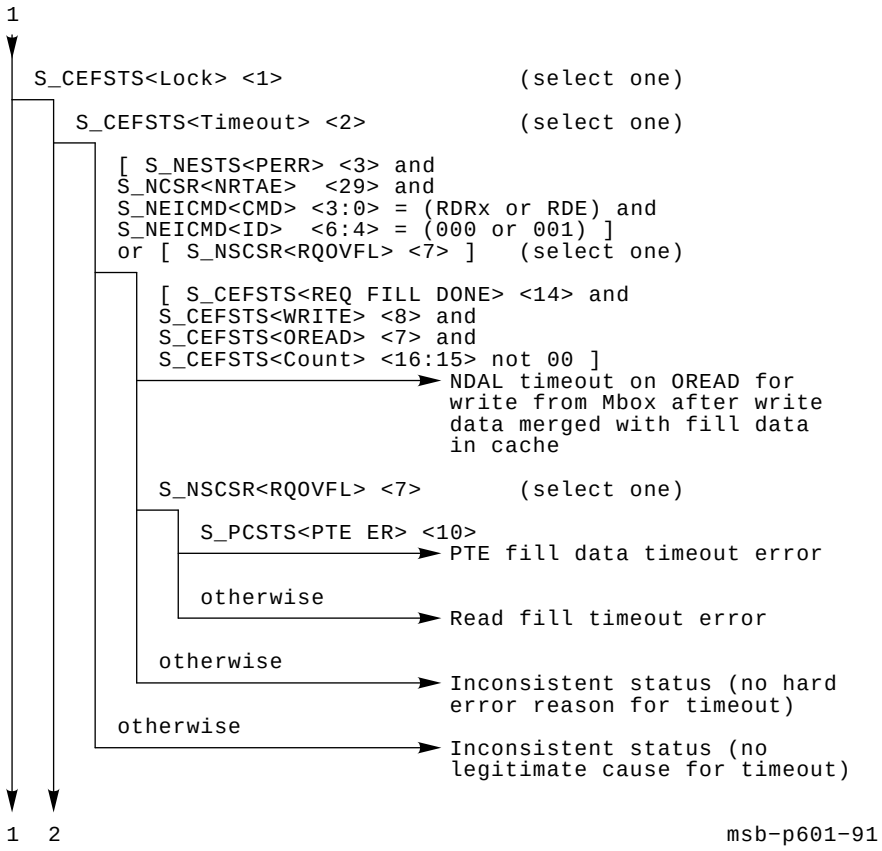


Figure 4–61 Cont'd on next page

Figure 4–61 (Cont.): Hard Error Interrupt Parse Tree

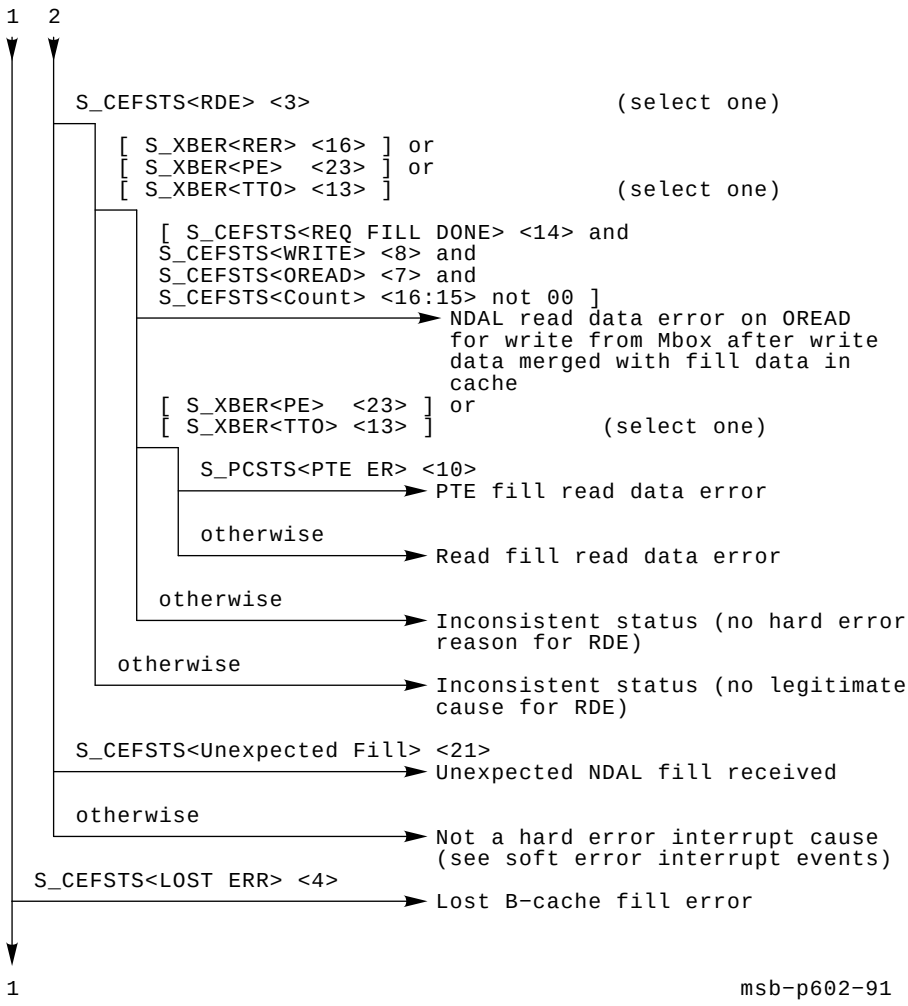


Figure 4–61 Cont'd on next page

Figure 4–61 (Cont.): Hard Error Interrupt Parse Tree

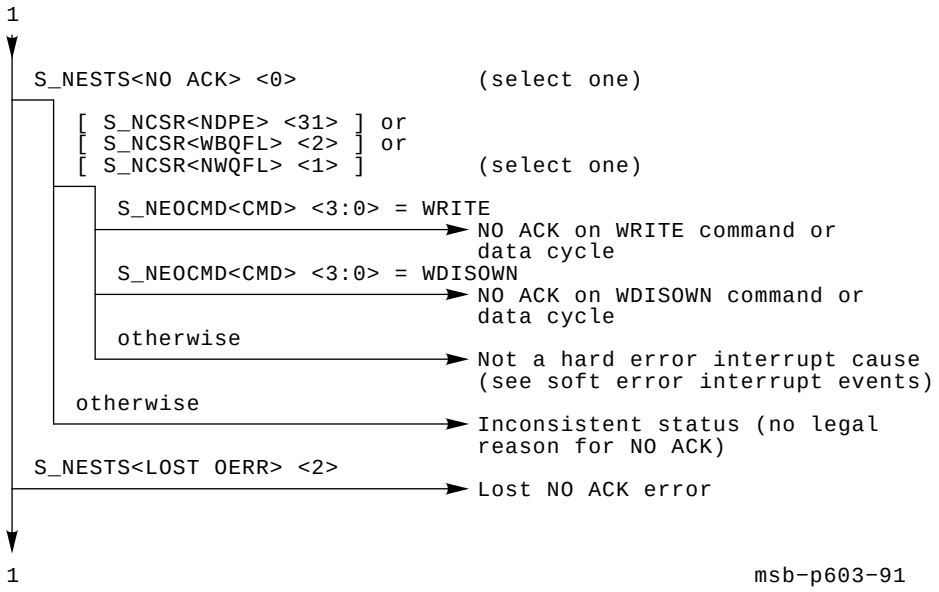


Figure 4–61 Cont'd on next page

Figure 4–61 (Cont.): Hard Error Interrupt Parse Tree

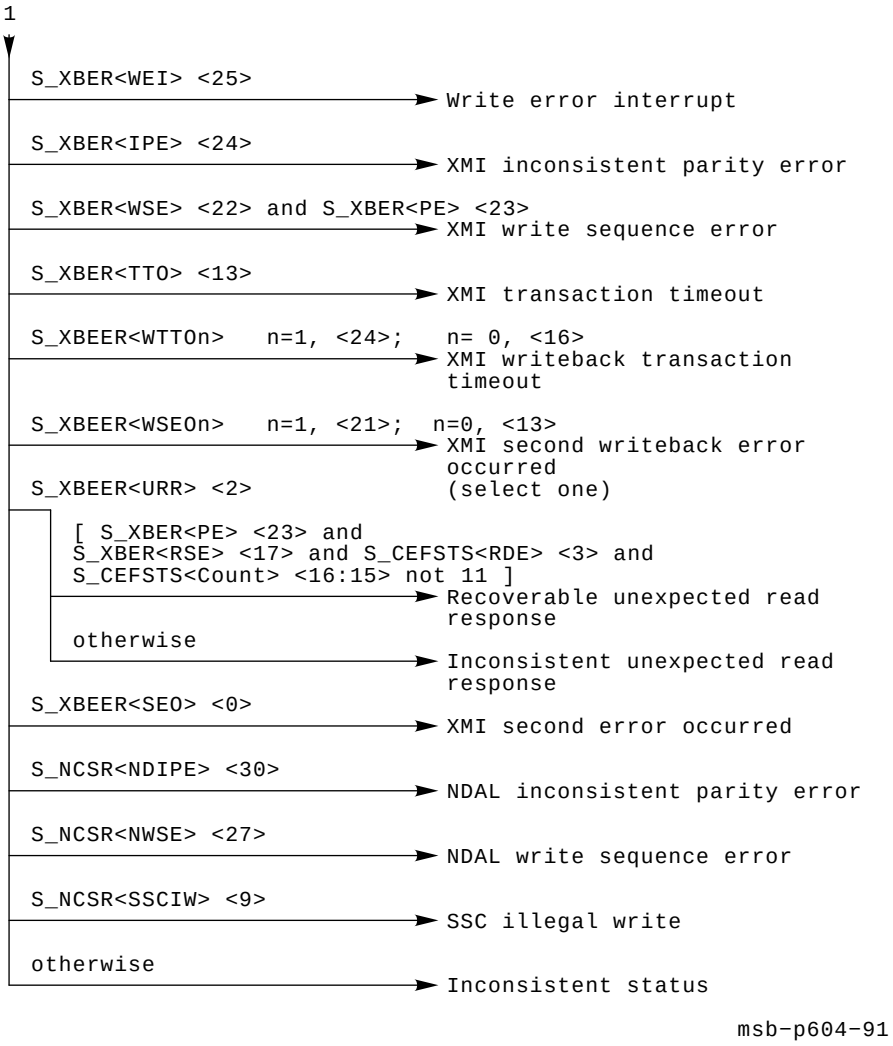


Figure 4–62: Soft Error Interrupt Parse Tree

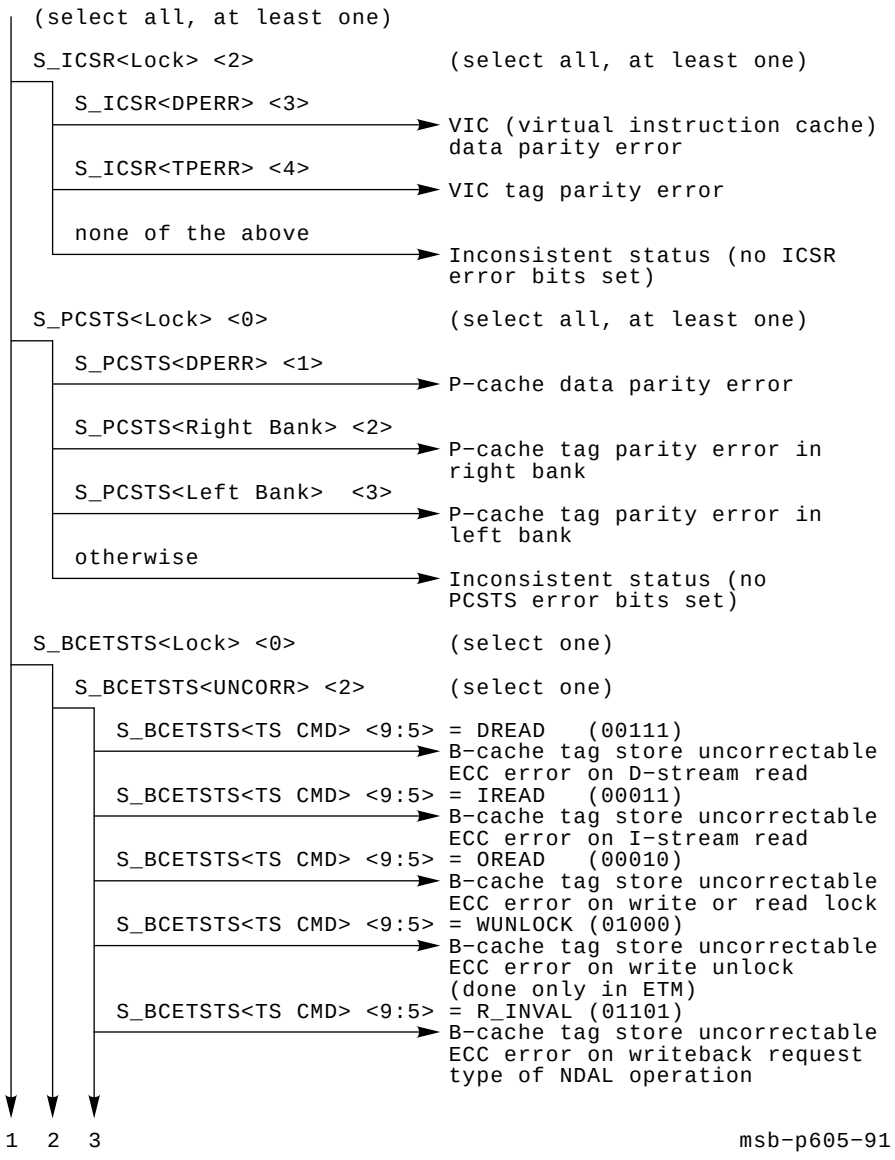


Figure 4–62 Cont'd on next page

Figure 4–62 (Cont.): Soft Error Interrupt Parse Tree

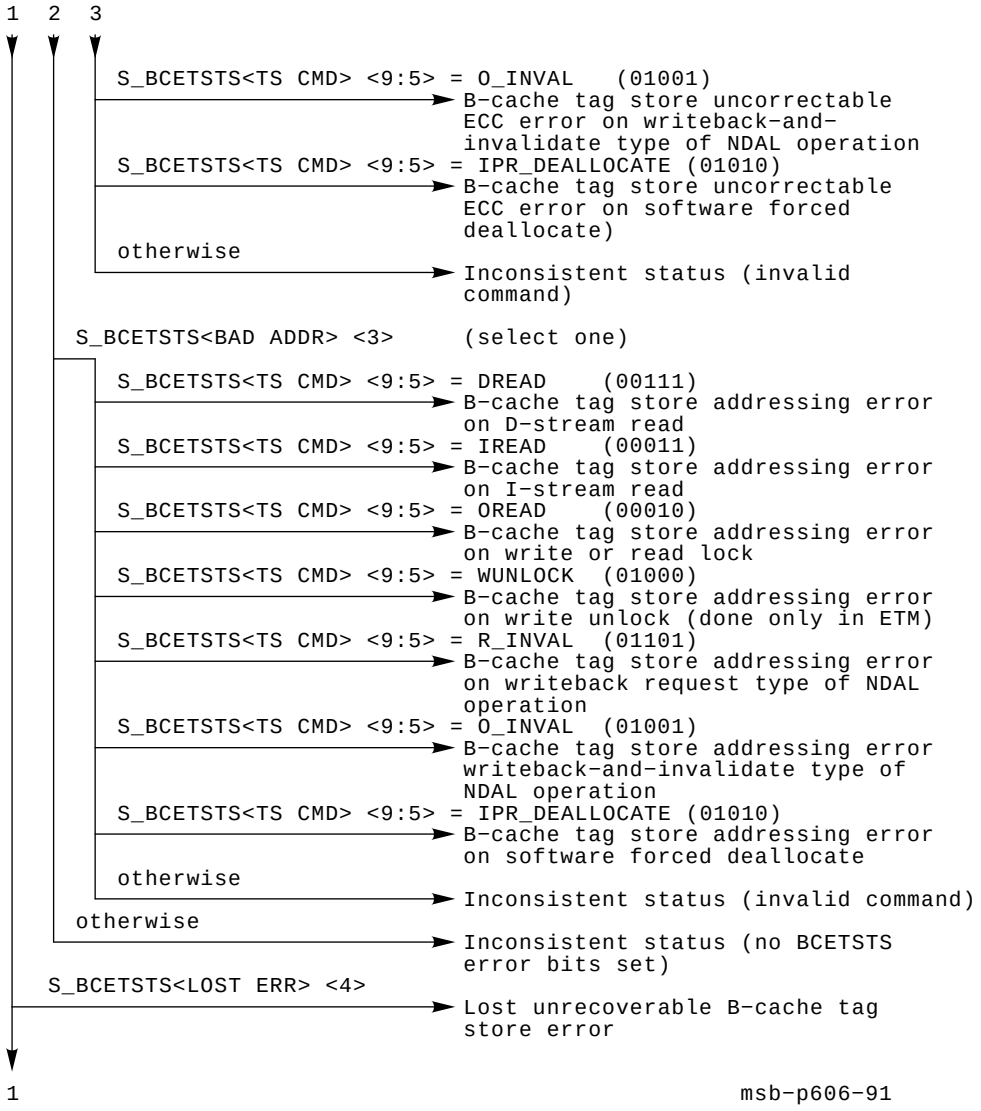


Figure 4–62 Cont'd on next page

Figure 4–62 (Cont.): Soft Error Interrupt Parse Tree

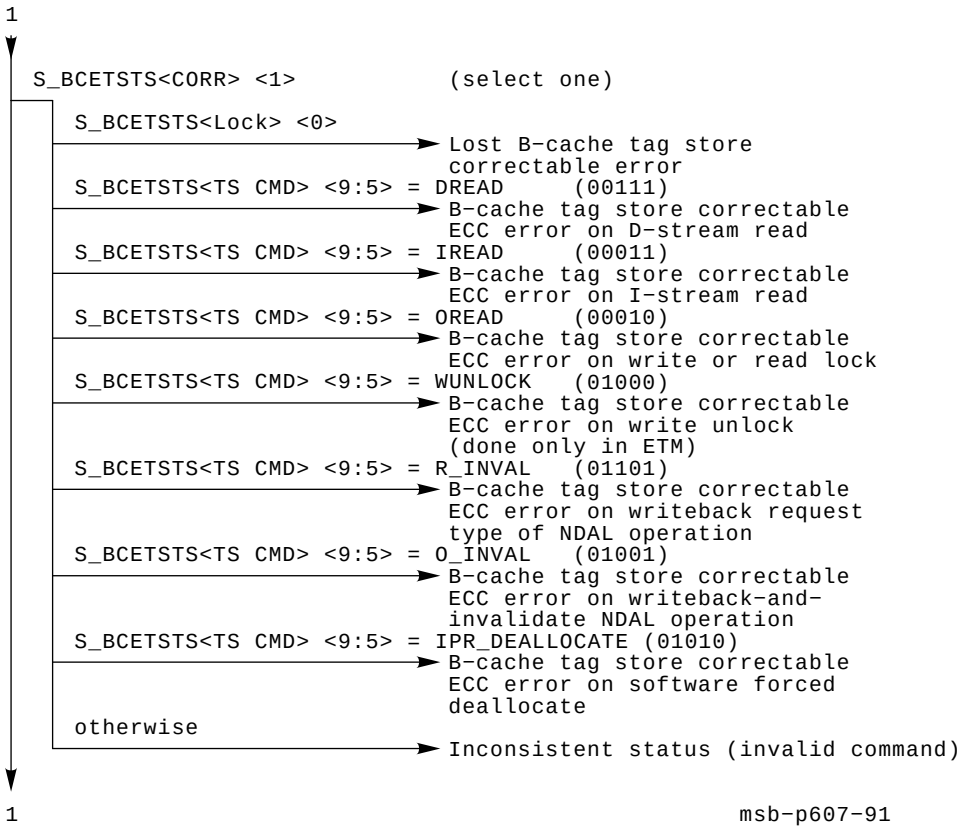


Figure 4–62 Cont'd on next page

Figure 4–62 (Cont.): Soft Error Interrupt Parse Tree

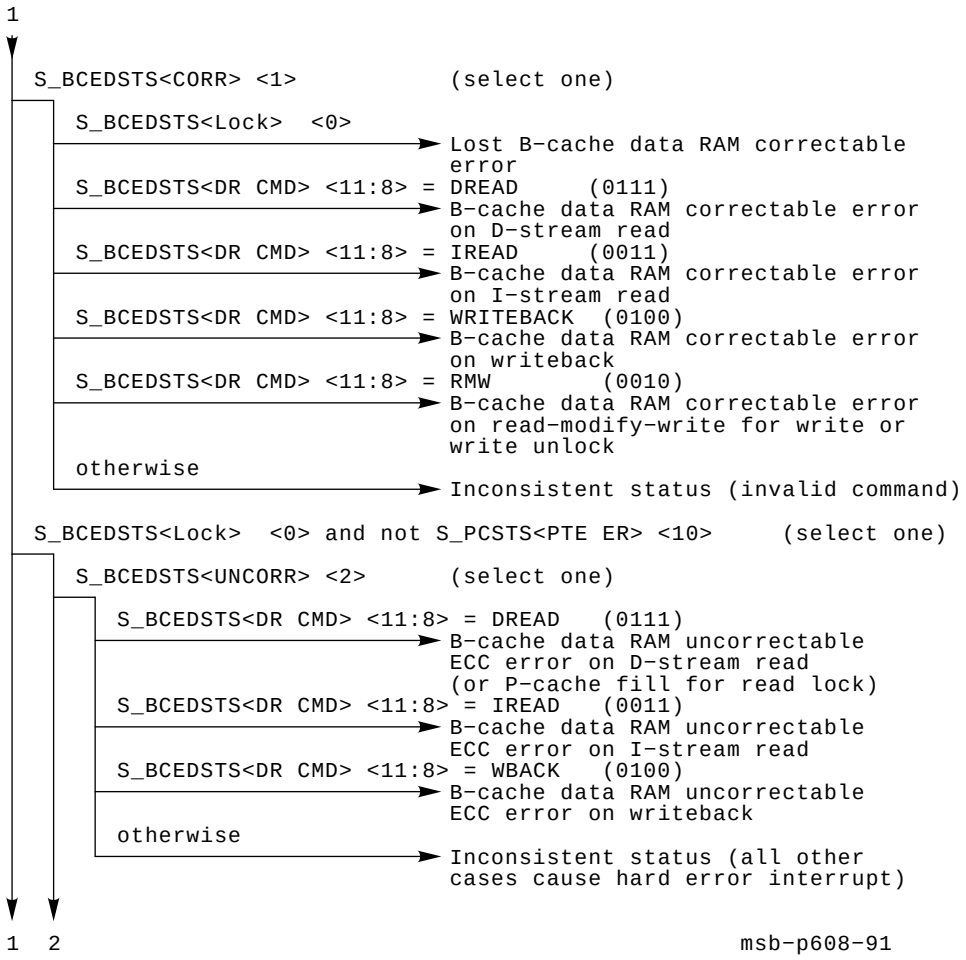


Figure 4–62 Cont'd on next page

Figure 4–62 (Cont.): Soft Error Interrupt Parse Tree

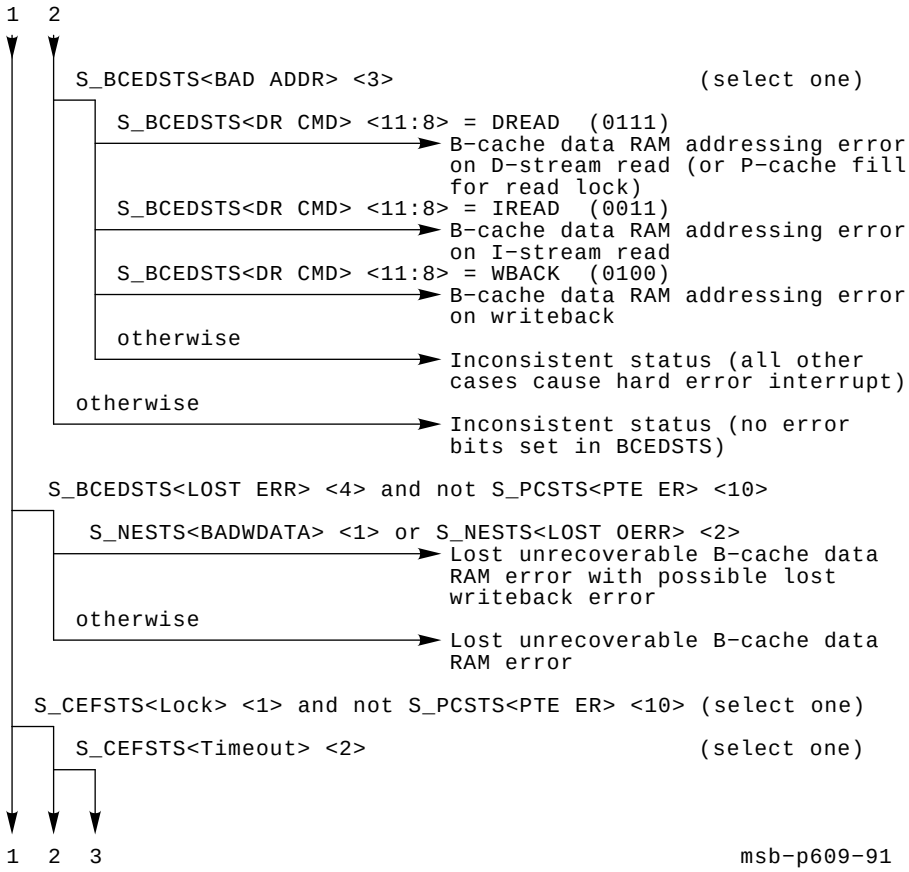


Figure 4–62 Cont'd on next page

Figure 4–62 (Cont.): Soft Error Interrupt Parse Tree

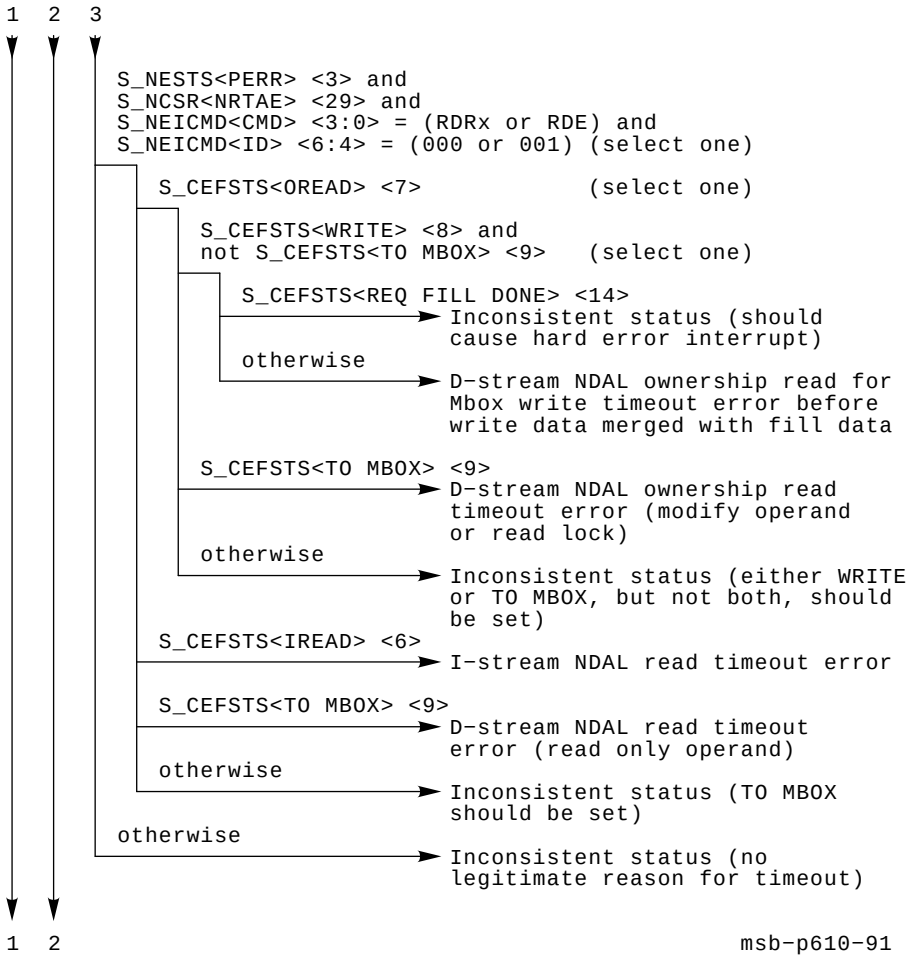


Figure 4–62 Cont'd on next page

Figure 4–62 (Cont.): Soft Error Interrupt Parse Tree

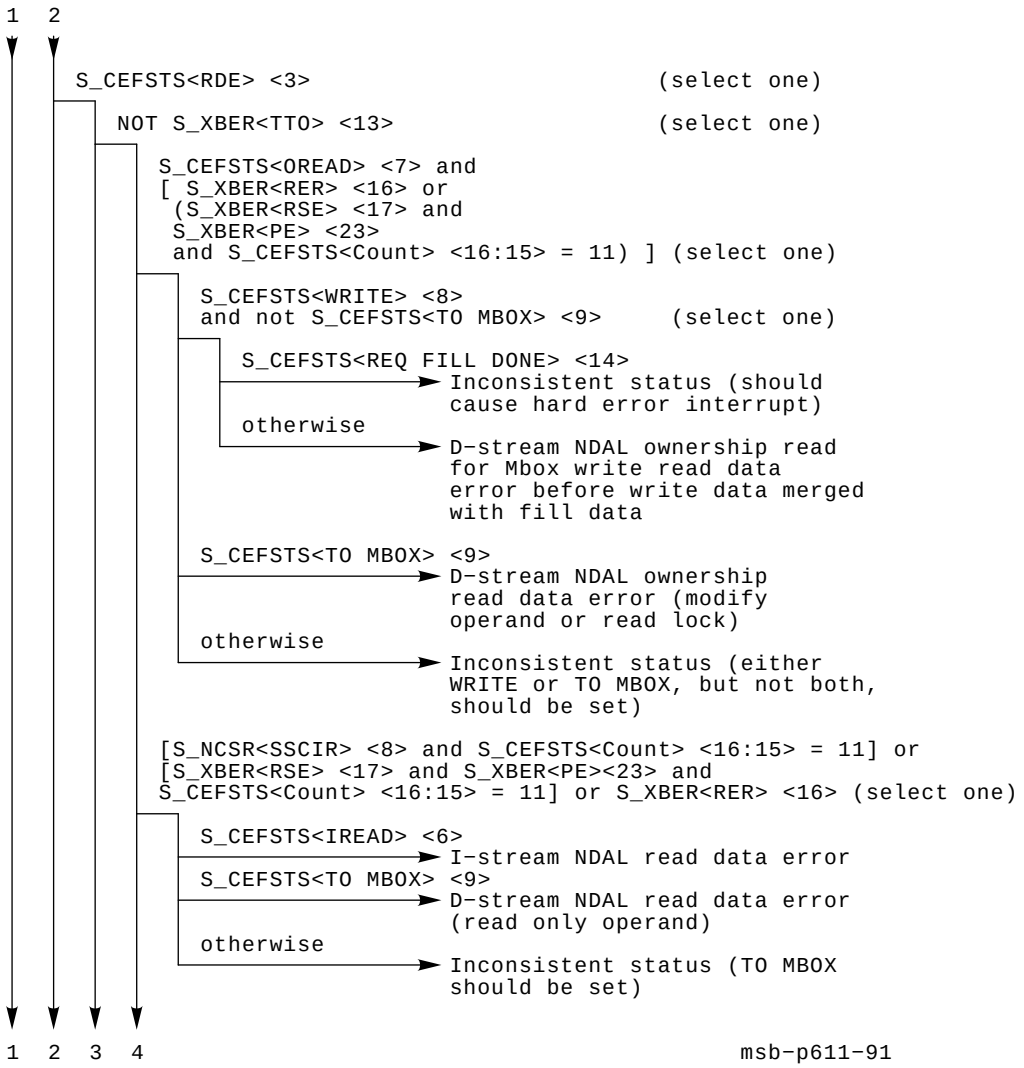


Figure 4–62 Cont'd on next page

Figure 4–62 (Cont.): Soft Error Interrupt Parse Tree

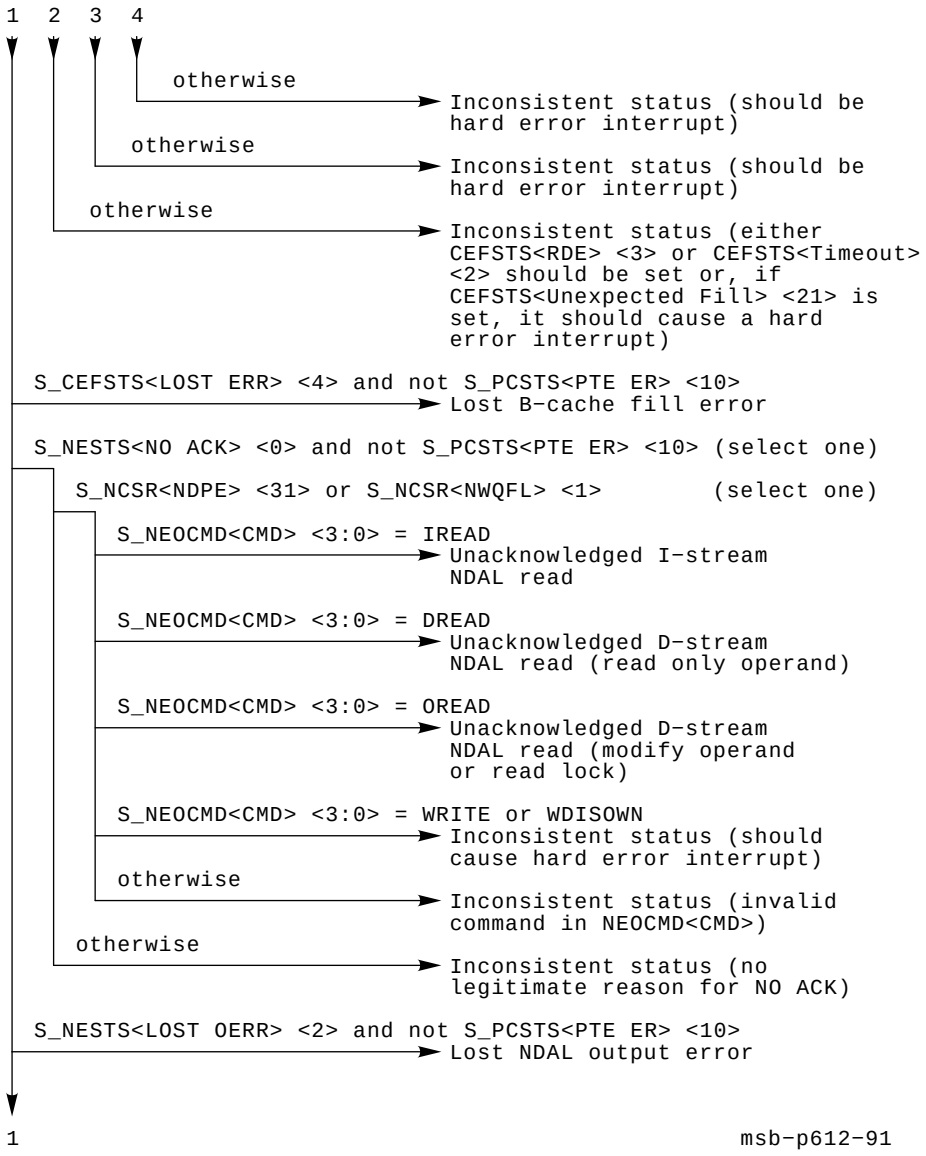


Figure 4–62 Cont'd on next page

Figure 4–62 (Cont.): Soft Error Interrupt Parse Tree

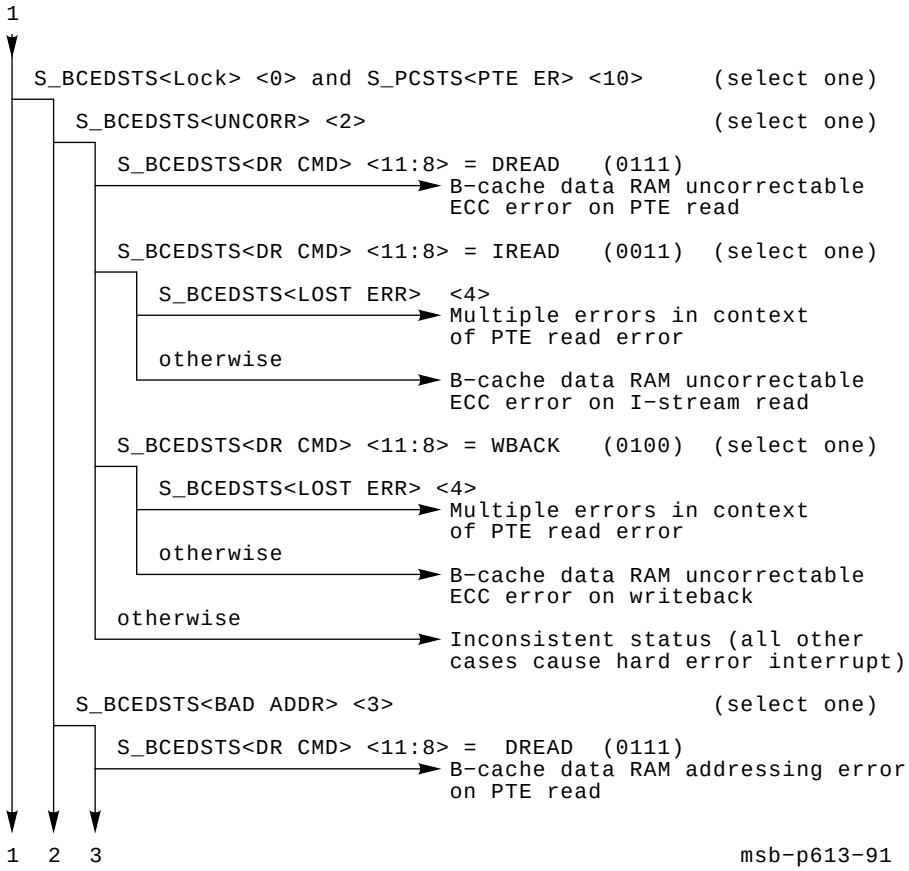


Figure 4–62 Cont'd on next page

Figure 4–62 (Cont.): Soft Error Interrupt Parse Tree

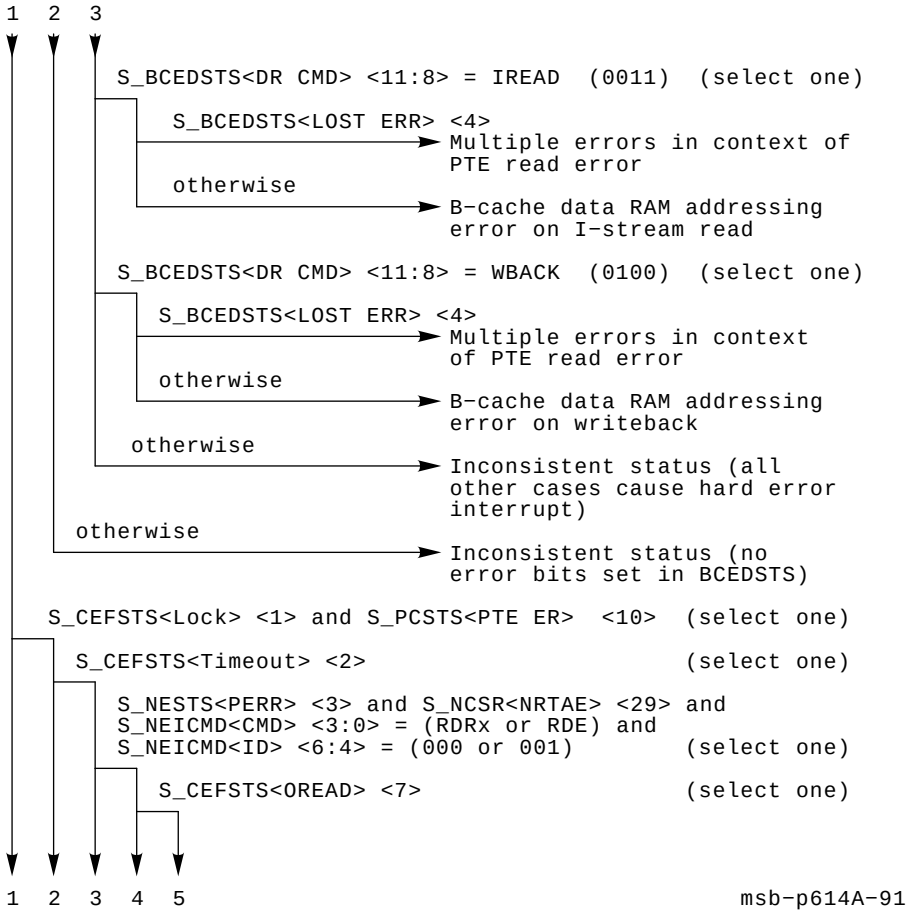


Figure 4–62 Cont'd on next page

Figure 4–62 (Cont.): Soft Error Interrupt Parse Tree

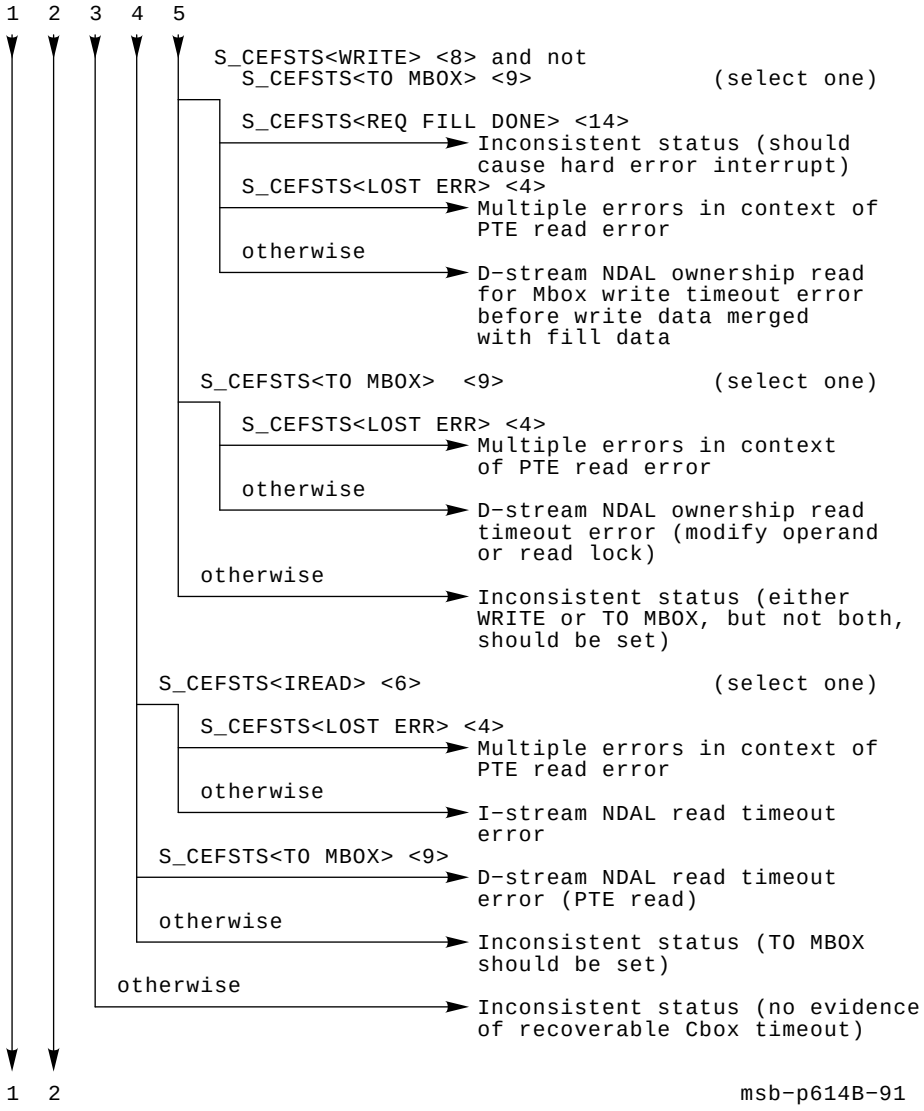


Figure 4–62 Cont'd on next page

Figure 4–62 (Cont.): Soft Error Interrupt Parse Tree

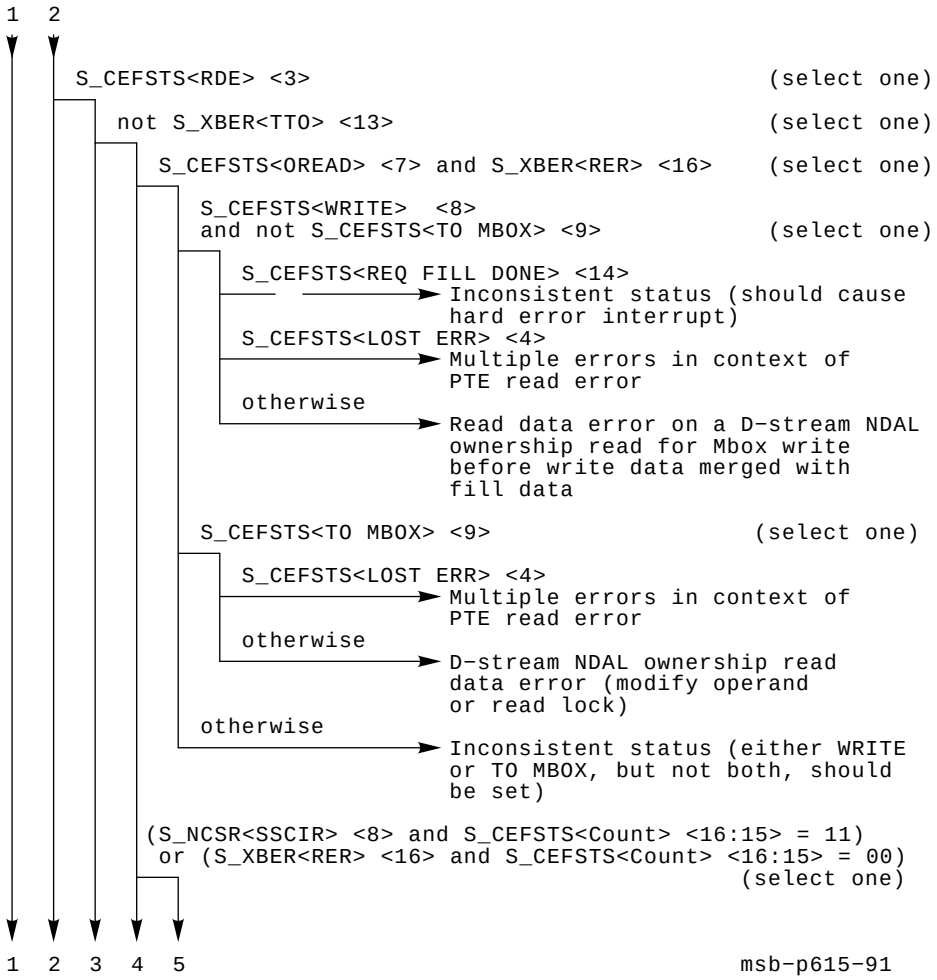


Figure 4–62 Cont'd on next page

Figure 4–62 (Cont.): Soft Error Interrupt Parse Tree

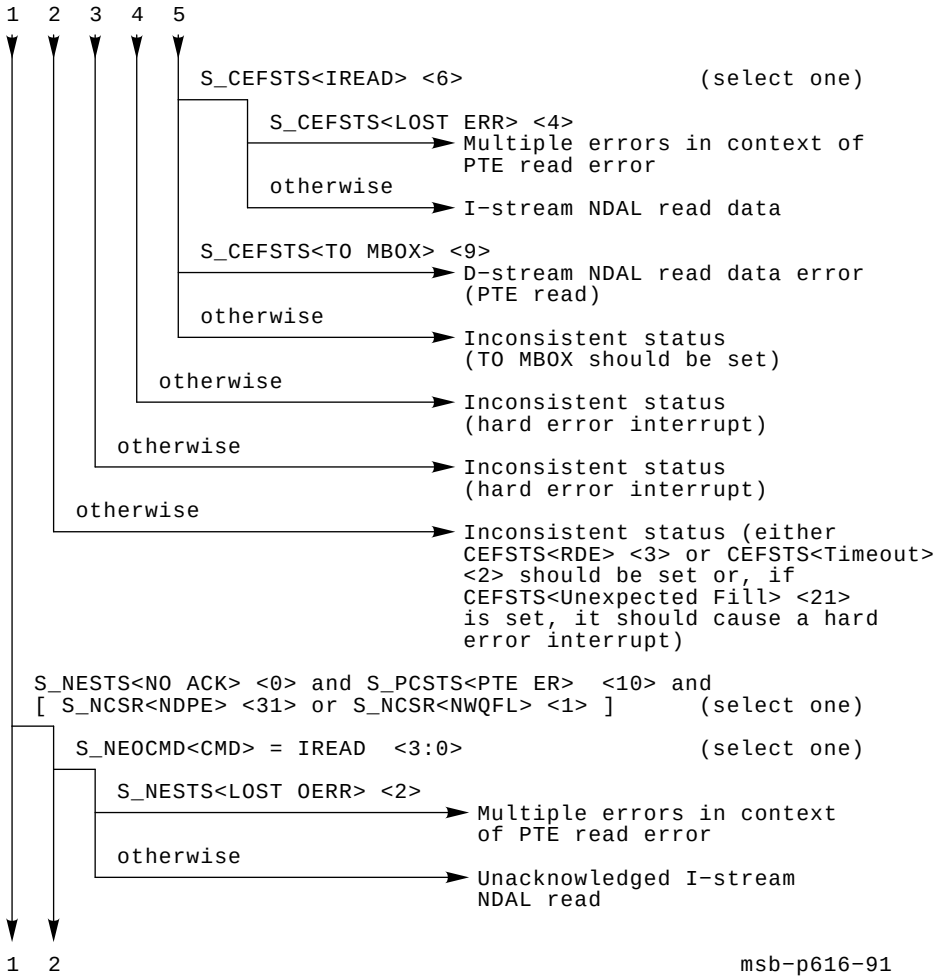
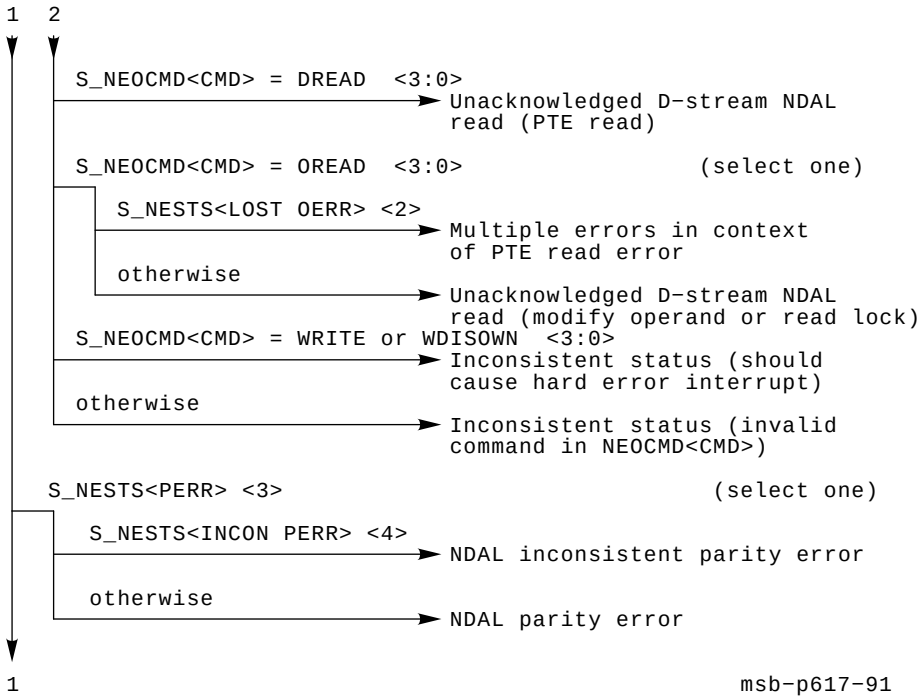


Figure 4–62 Cont'd on next page

Figure 4–62 (Cont.): Soft Error Interrupt Parse Tree



Chapter 5
MS65A Memory Registers

Table 5–1: MS65A Memory Control and Status Registers

Name	Mnemonic	Address
Device Register	XDEV	BB[1] + 00
Bus Error Register	XBER	BB + 04
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
Block State ECC Error Register	BECER	BB + 38
Block State ECC Address Register	BECEA	BB + 3C
Starting Address Register	STADR	BB + 50
Ending Address Register	ENADR	BB + 54
Segment/Interleave Control Register	INTLV	BB + 58
Memory Control Register 3	MCTL3	BB + 5C
Memory Control Register 4	MCTL4	BB + 60
Block State Control Register	BSCTL	BB + 68
Block State Address Register	BSADR	BB + 6C
EEPROM Control Register	EECTL	BB + 70
Timeout Control/Status Register	TMOER	BB + 74

[1] "BB" refers to the base address of an XMI node (E180 0000 + (node ID x 8000)).

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

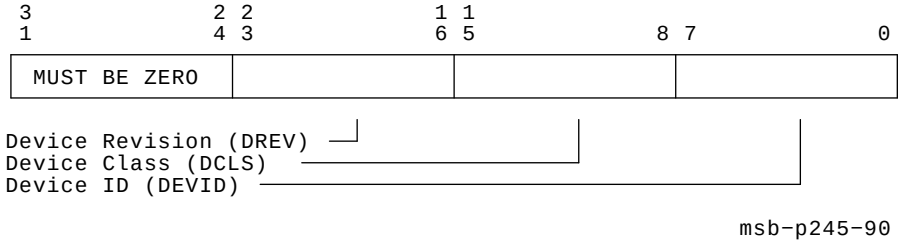


Figure 5–2: Bus Error Register (XBER)
BB + 04

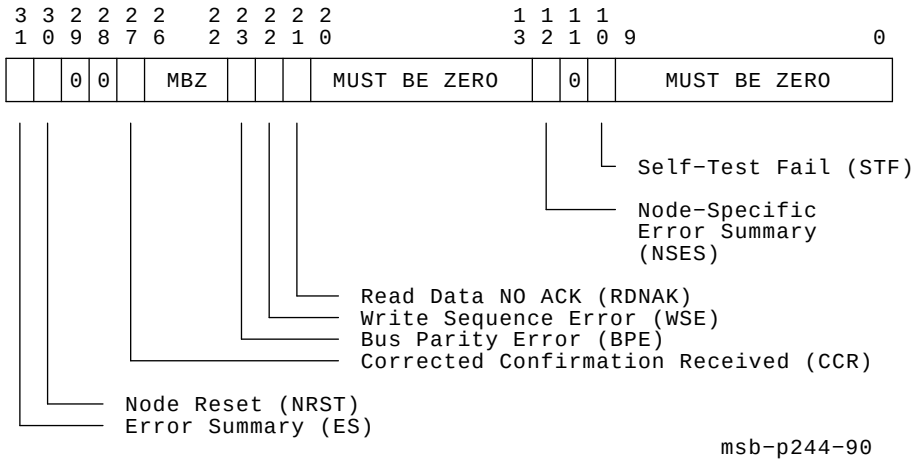
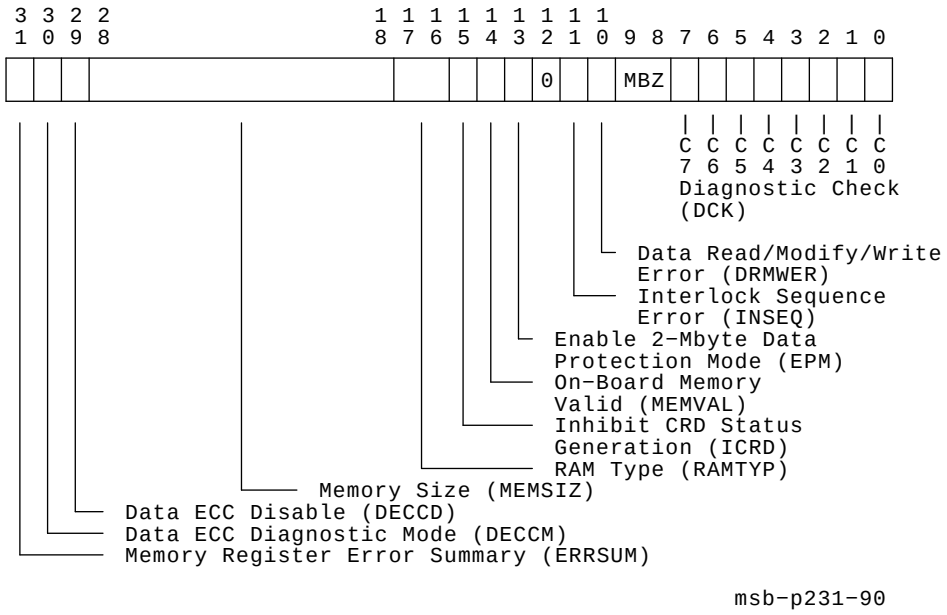
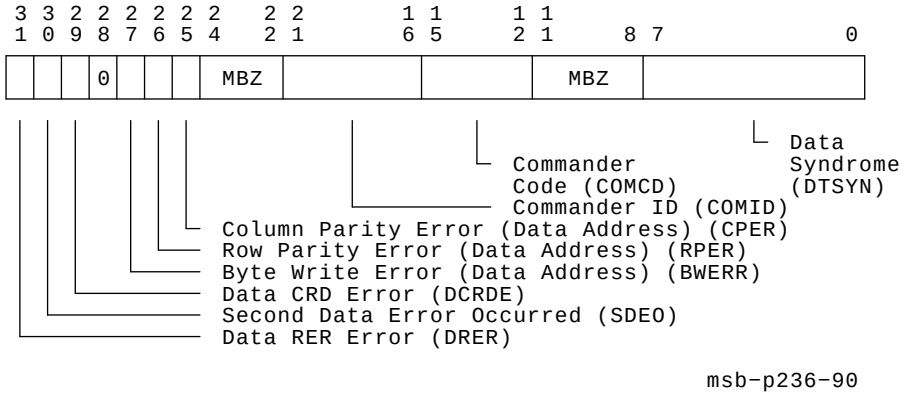


Figure 5–3: Memory Control Register 1 (MCTL1)
BB + 14



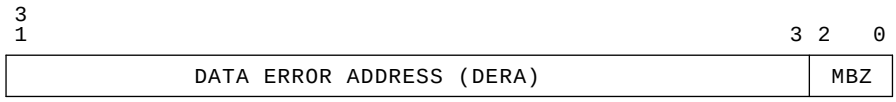
msb–p231–90

Figure 5–4: Memory ECC Error Register (MECER)
BB + 18



msb–p236–90

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



msb–p235–90

Figure 5–6: Memory Control Register 2 (MCTL2)
BB + 30

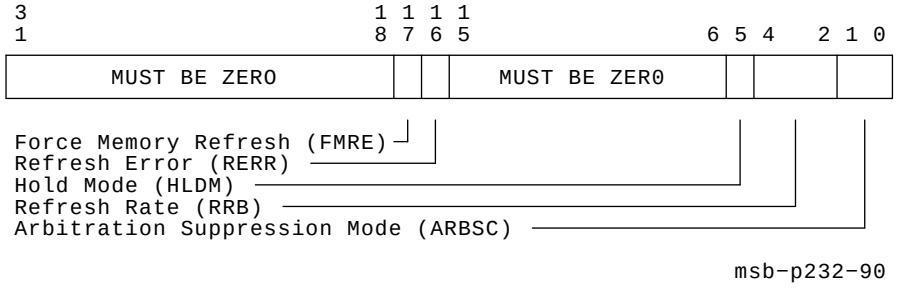


Figure 5–7: TCY Tester Register (TCY)
BB + 34

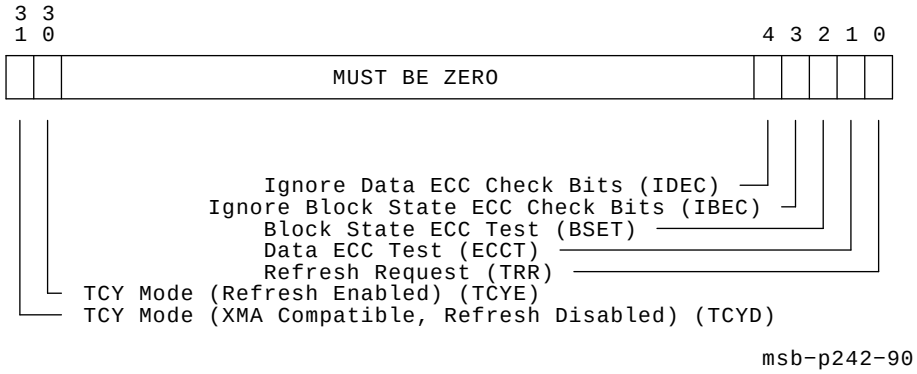


Figure 5–8: Block State ECC Error Register (BECER)
BB + 38

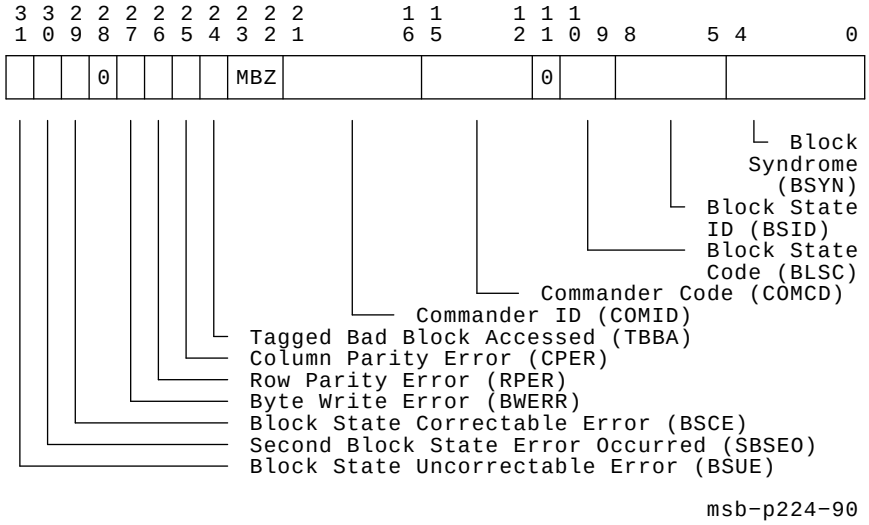


Figure 5–9: Block State ECC Address Register (BECEA)
BB + 3C

Figure 5–10: Starting Address Register (STADR)
BB + 50

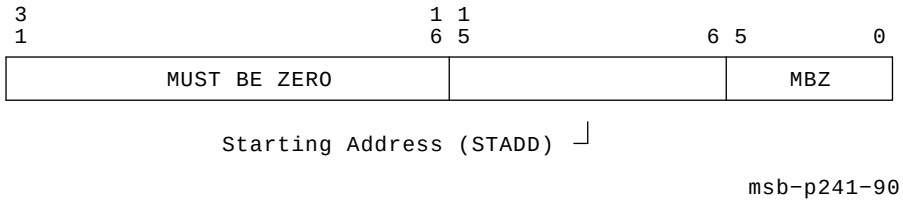


Figure 5–11: Ending Address Register (ENADR)
BB + 54

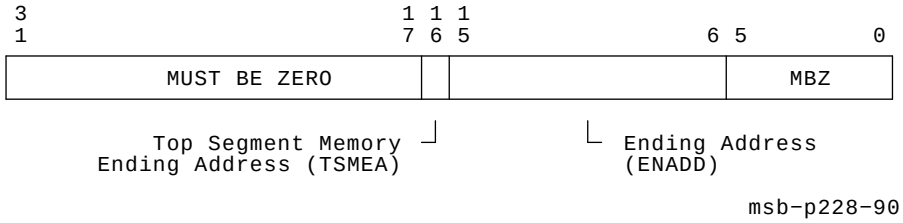


Figure 5–12: Segment/Interleave Register (INTLV)
BB + 58

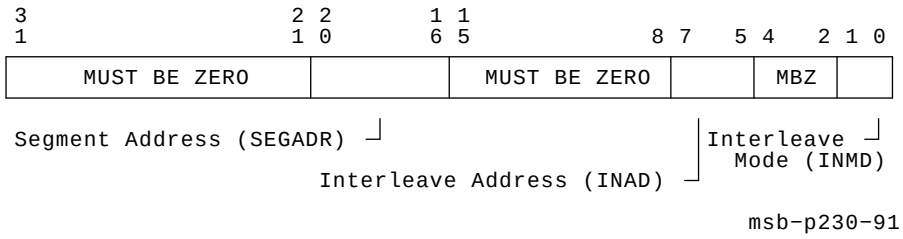
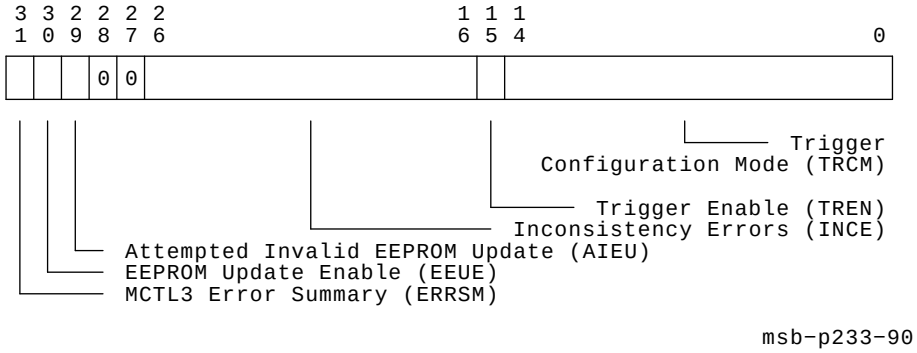
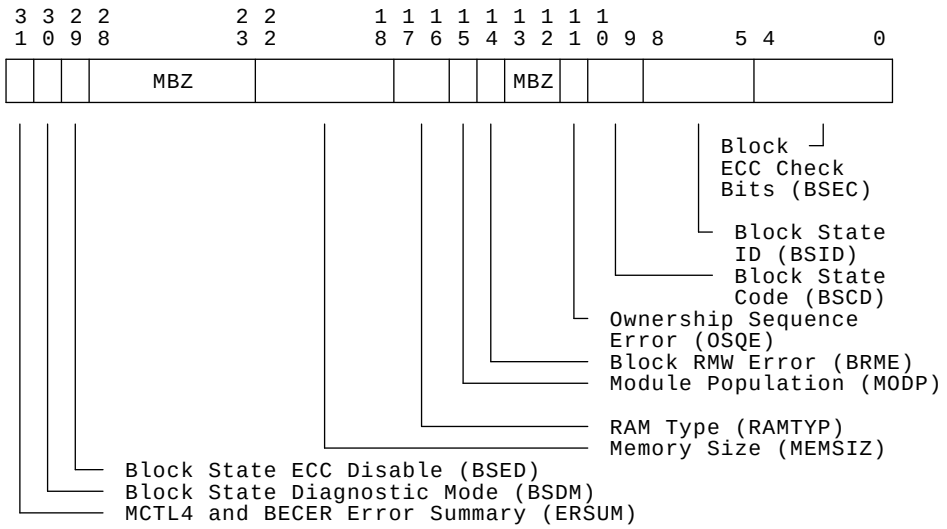


Figure 5–13: Memory Control Register 3 (MCTL3)
BB + 5C



msb–p233–90

Figure 5–14: Memory Control Register 4 (MCTL4)
BB + 60



msb–p234–90

Figure 5–15: Block State Control Register (BSCTL)
BB + 68

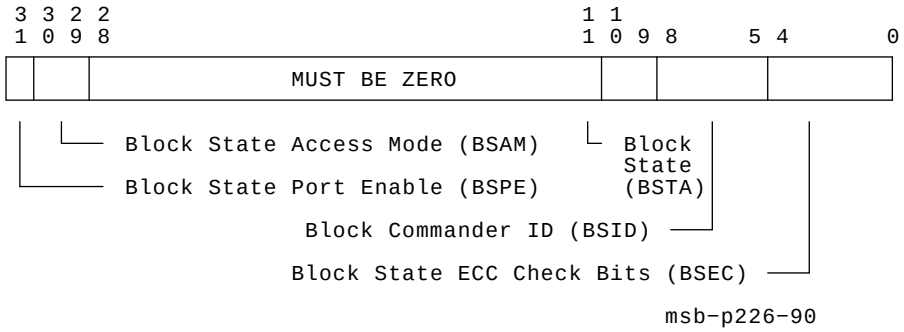


Figure 5–16: Block State Address Register (BSADR)
BB + 6C

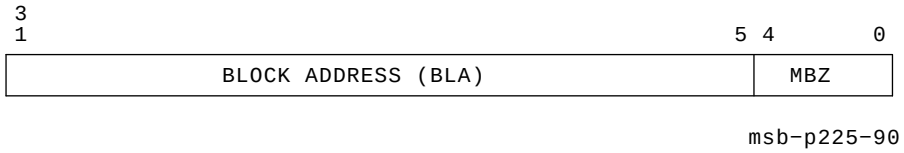


Figure 5–17: EEPROM Control Register (EECTL)
BB + 70

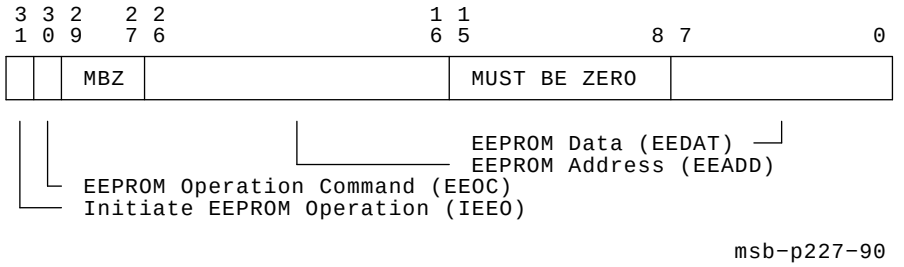
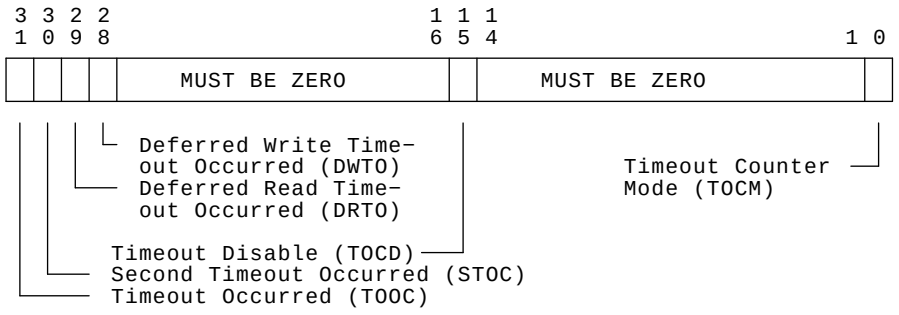


Figure 5–18: Timeout Control/Status Register (TMOER) BB + 74



Chapter 6
DWMBB Adapter Registers

The DWMBB 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 DWMBB registers: some of which are XMI required registers, some DWMBB/A registers, some DWMBB/B registers, and the VAXBI Device Register for the DWMBB/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: DWMBB 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
DWMBB/A Error Summary Register	AESR	BB + 10
Interrupt Mask Register	AIMR	BB + 14
Implied Vector Interrupt Destination/Diagnostic Register	AIVINTR	BB + 18
Diagnostic 1 Register	ADG1	BB + 1C
Utility Register	AUTLR	BB + 20
Control and Status Register	ACSR	BB + 24
Return Vector Register	ARVR	BB + 28
Failing Address Extension Register	XFAER	BB + 2C
VAXBI Error Address Register	ABEAR	BB + 30
Control and Status Register	BCSR	BB + 40
DWMBB/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

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

X=XMI register, resides on the DWMBB/A module
A=Resides on the DWMBB/A module
B=Resides on the DWMBB/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*.

Table 6–1 (Cont.): DWMBB Registers

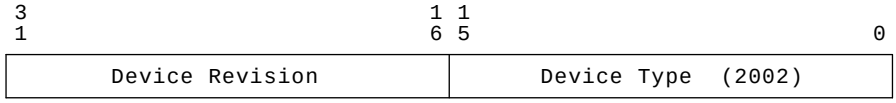
Name	Mnemonic[1]	Address[2]
Diagnostic Control Register 1	BDCR1	BB + 58
Reserved Register	–	BB + 5C
Page Map Register (first location)	PMR	BB + 200
.	.	.
.	.	.
Page Map Register (last location)	PMR	BB + 401FC
Device Register[3]	DTYPE	bb + 00

Table 6–2: XMI Required Registers

Name	Mnemonic	Address[1]
Device Register	XDEV	BB + 00
Bus Error Register	XBER	BB + 04
Failing Address Register	XFADR	BB + 08
Failing Address Extension Register	XFAER	BB + 2C

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

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



msb-p100-89

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

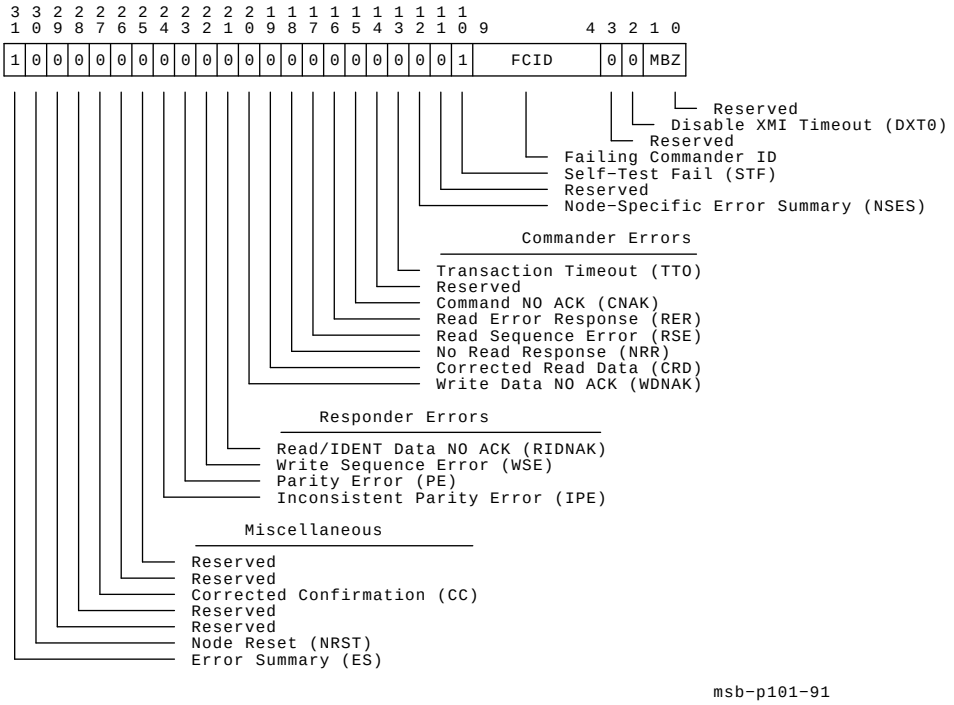
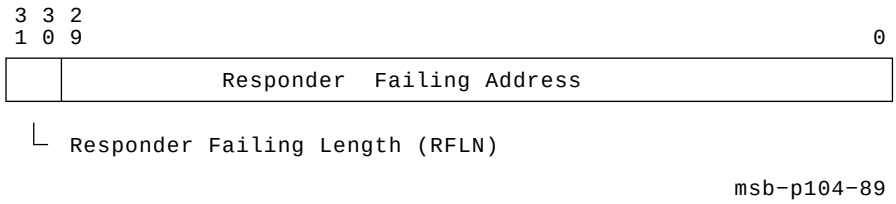


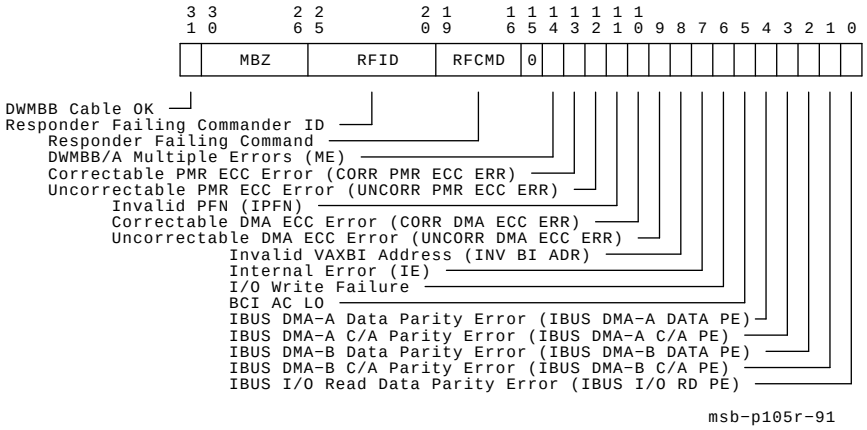
Figure 6–3: Failing Address Register (XFADR)
BB + 08

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



msb–p104–89

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



msb–p105r–91

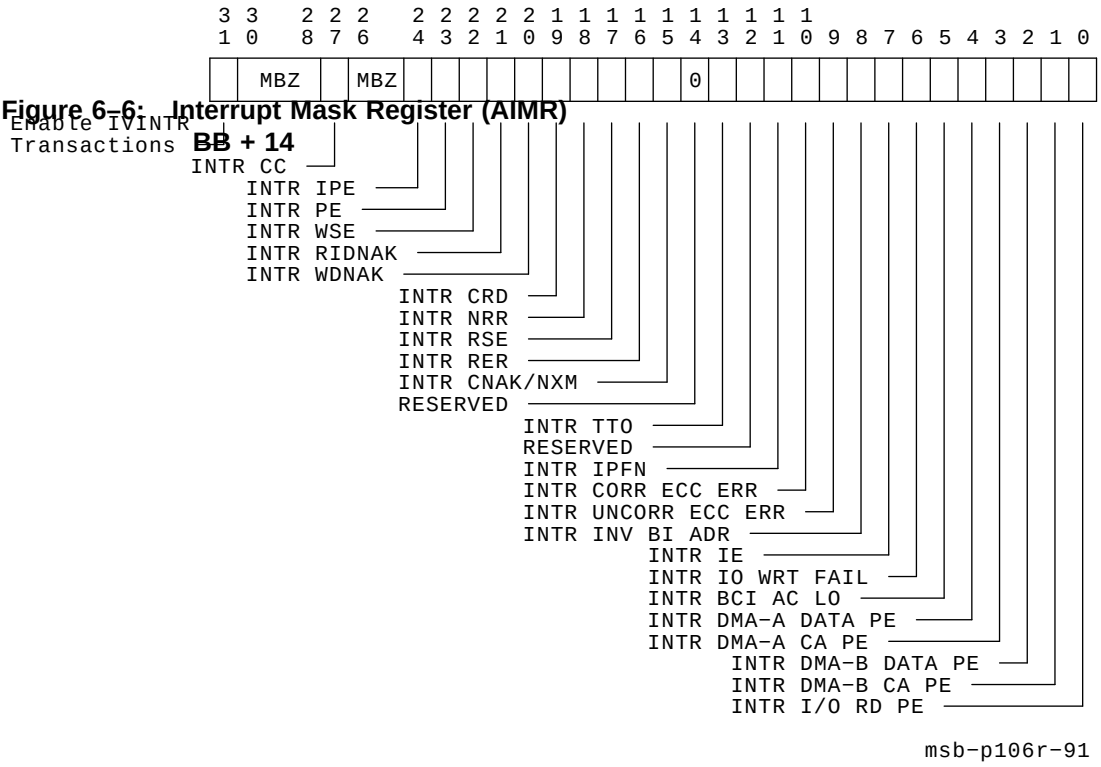
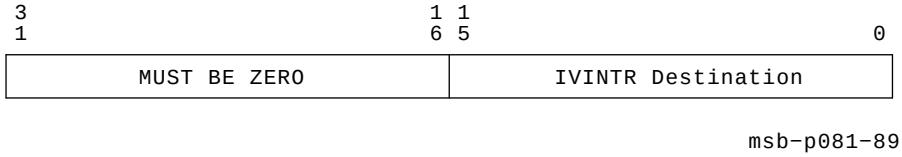


Figure 6–6: Interrupt Mask Register (AIMR) BB + 14

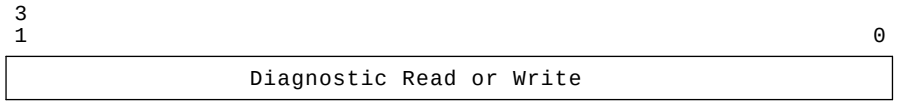
Figure 6–7: Implied Vector Interrupt Destination/Diagnostic Register (AIVINTR) BB + 18



AIVINTR, when used during XBI-initiated IVINTR transactions.

Figure 6–7 Cont'd on next page

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



msb−p080−89

AIVINTR, when used during diagnostics.

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

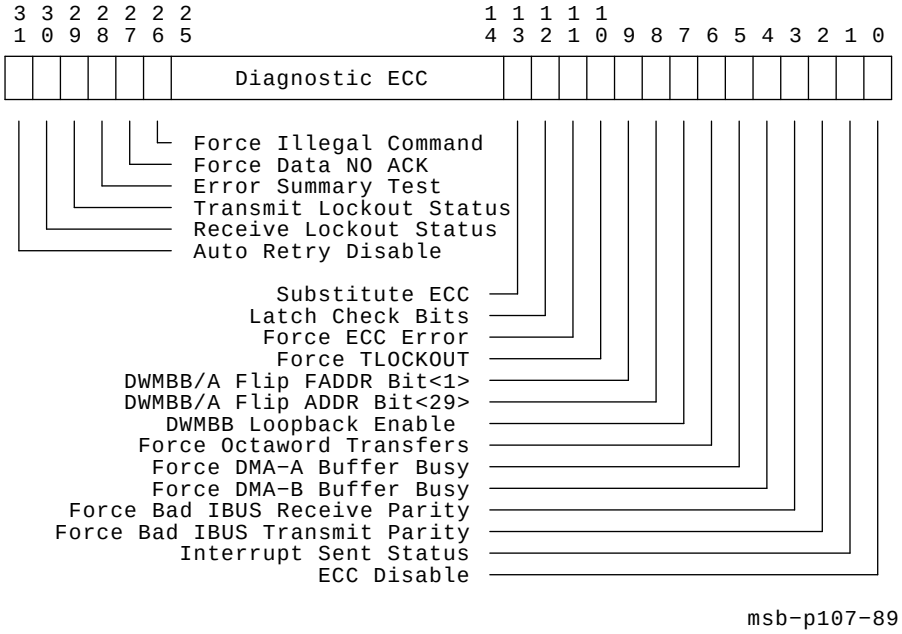


Figure 6–9: Utility Register (AUTLR)
BB + 20

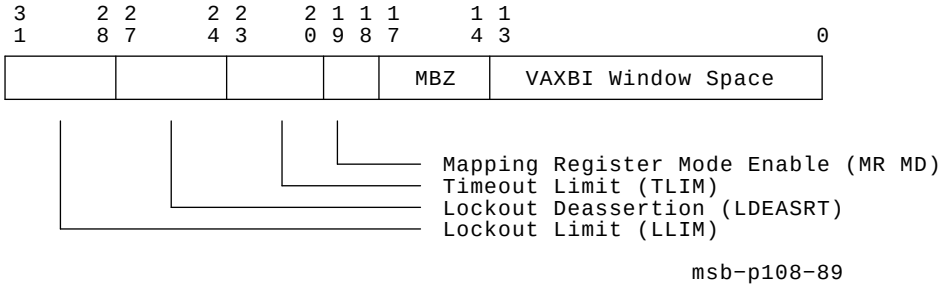
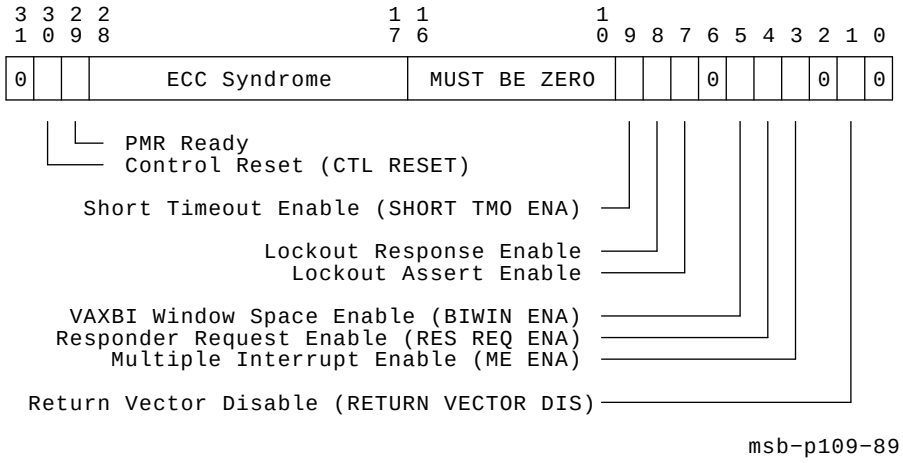
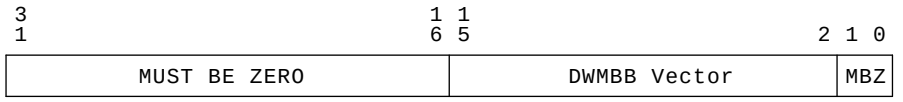


Figure 6–10: Control and Status Register (ACSR)
BB + 24



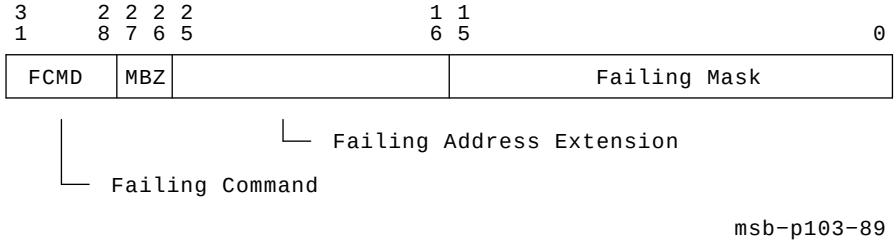
msb–p109–89

Figure 6–11: Return Vector Register (ARVR)
BB + 28



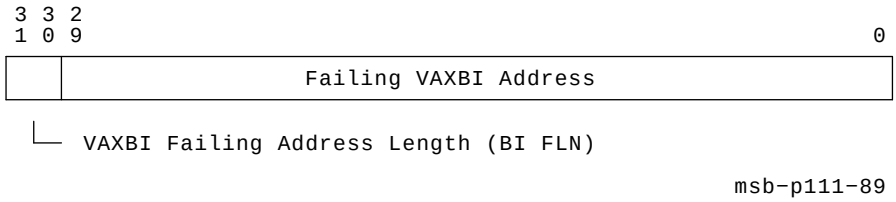
msb–p110–89

Figure 6–12: Failing Address Extension Register (XFAER)
BB + 2C



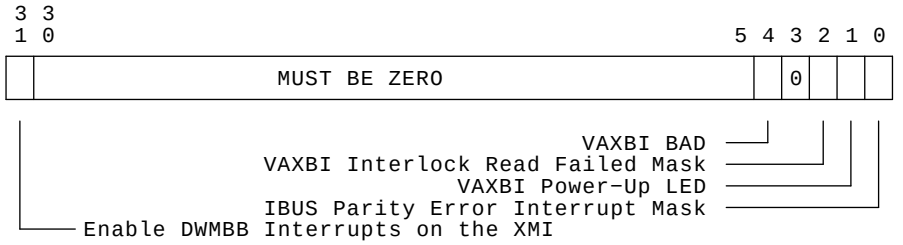
msb–p103–89

Figure 6–13: VAXBI Error Address Register (ABEAR)
BB + 30



msb–p111–89

Figure 6–14: Control and Status Register (BCSR)
BB + 40



msb–p113–89

Figure 6–15: DWMBB/B Error Summary Register (BESR)
BB + 44

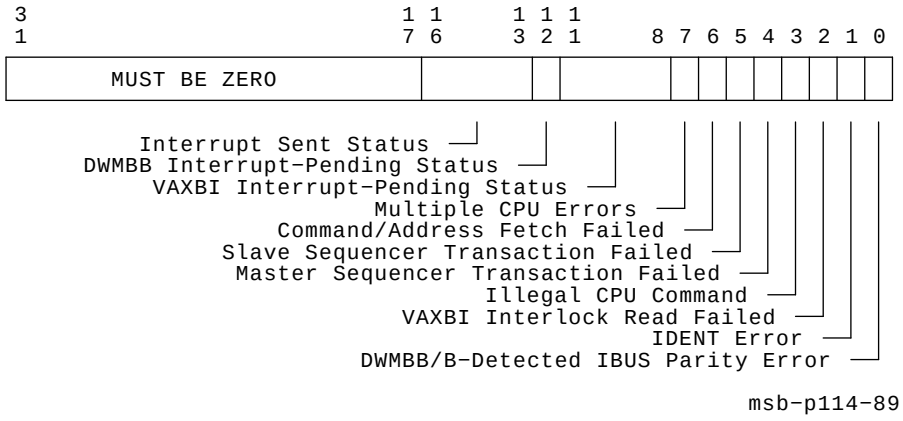


Figure 6–16: Interrupt Destination Register (BIDR)
BB + 48

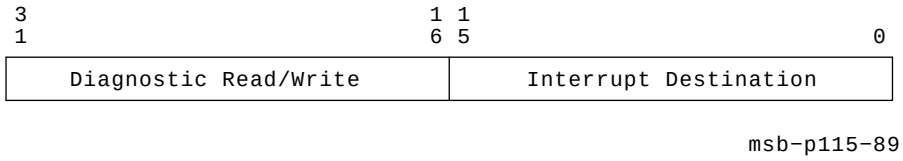


Figure 6–17: Timeout Address Register (BTIM)
BB + 4C

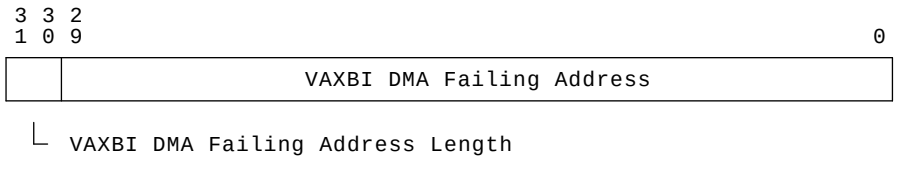


Figure 6–18: Vector Offset Register (BVOR)
BB + 50

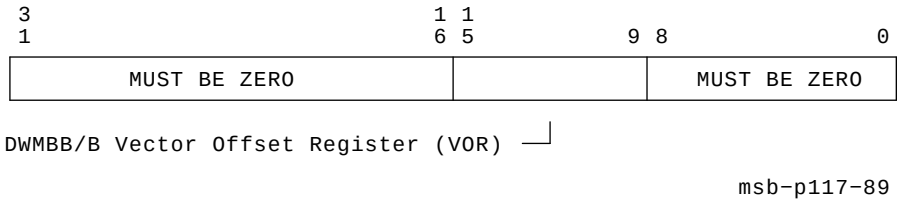


Figure 6–19: Vector Register (BVR)
BB + 54

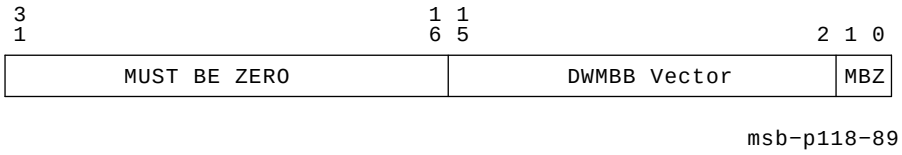


Figure 6–20: Diagnostic Control Register 1 (BDCR1)
BB + 58

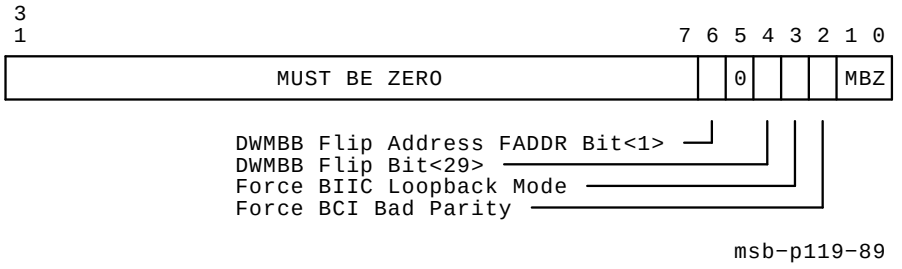


Figure 6–21: Page Map Register (PMR)
BB + 200 to BB + 401FC

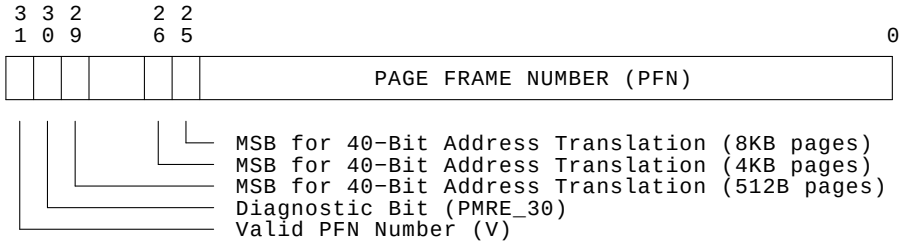
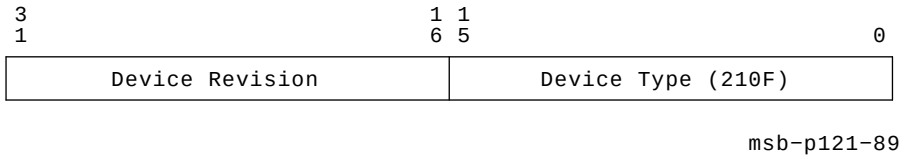


Figure 6–22: VAXBI Device Register (DTYPE)
bb + 00



Index

A

B

C

D

E

F

I

L

M

N

O

P

R

S

T

U

V

W

X