UNISYS

XE500
BTOS
Administration Guide

Relative to Release
Level 7.0

Priced Item

May 1988
Distribution Code SA
Printed in U S America
5028608

UNISYS

XE500 BTOS Administration Guide

Relative to Release
Level 7.0

Priced Item

May 1988
Distribution Code SA
Printed in U S America
5028608

The names, places and/or events used in this publication are not intended to correspond to any individual, group, or association existing, living or otherwise. Any similarity or likeness of the names, places, and/or events with the names of any individual living or otherwise, or that of any group or association is purely coincidental and unintentional.

NO WARRANTIES OF ANY NATURE ARE EXTENDED BY THE DOCUMENT. Any product and related material disclosed herein are only furnished pursuant and subject to the terms and conditions of a duly executed Program Product License or Agreement to purchase or lease equipment. The only warranties made by Unisys, if any, with respect to the products described in this document are set forth in such License or Agreement. Unisys cannot accept any financial or other responsibility that may be the result of your use of the information in this document or software material, including direct, indirect, special or consequential damages.

You should be very careful to ensure that the use of this information and/or software material complies with the laws, rules, and regulations of the jurisdictions with respect to which it is used.

The information contained herein is subject to change without notice. Revisions may be issued to advise of such changes and/or additions.

Comments or suggestions regarding this document should be submitted on a User Communication Form (UCF) with the CLASS specified as 2 (S.S.W.:System Software), the Type specified as 1, and the product specified as the 7-digit form number of the manual (for example, 5028608).

About This Guide

This guide explains how to administer the XEBTOS operating system after it has been installed.

Who Should Use This Guide

This guide will help you if you are an experienced user of XEBTOS or an experienced administrator of another operating system. You should be familiar with the *XE500 BTOS Operations Guide*.

How to Use This Guide

If you are performing XEBTOS administrative tasks for the first time, you should read sections 1 and 2. They contain basic information you will need for understanding your role and executing master commands.

By scanning the contents and reviewing the topics before you start, you may find this guide easier to use. To find definitions of unfamiliar words, use the glossary; to locate specific information, use the index.

In addition, you may want to review the *XE500 BTOS Operations Guide* for an overview of user operations.

How This Guide is Arranged

This guide is divided into sections, with related subjects grouped together. Section 2 explains how to execute master utilities using master commands. Thereafter, the general sequence of topics is as a typical administrator might find a need for them.

Conventions

The following conventions apply throughout this guide:

- Names of processor boards (such as CP) refer to both standard boards and X boards (such as CP–X).
- When two keys are used together for an operation, their names are hyphenated (for example, CODE–D).
- In the tape file formats, half–inch tape file names appear in the format

  [tapexy]m

  in which x, y, and m are variables. In this particular example, x, y, and m represent numerical values determined by a specific tape drive and file mark.
- In BTOS command forms, optional fields and parameters are enclosed in square brackets.
- Numbers are decimal unless they have the suffix h for hexidecimal (for example, 100h).

Reference Material

This guide contains two appendixes, a glossary, and an index.

- For definitions of the various types of status codes the system can report, refer to appendix A.
- For information on disk naming conventions, processor naming conventions, and board slot number conventions, refer to appendix B.
- For definitions of key terms used in this guide or related to the software, refer to the glossary.

Related Product Information

For information on creating customized versions of the operating systems that run on XE520 processors, refer to the *BTOS II Customizer Programming Guide*.

For a complete list of system status codes, with definitions and, if applicable, suggestions on how to recover, refer to the *BTOS II System Status Codes Reference Manual*.

For a description of how to use the Debugger, refer to the *XE500 BTOS Debugger Operations Guide*.

For information on installing and implementing XEBTOS system software, refer to the *XE500 BTOS Installation and Implementation Guide*.

For information on the XEBTOS tasks routinely performed by anyone using the XE520, refer to the *XE500 BTOS Operations Guide*.

For a description of the XEBTOS system software and XE520 hardware features, refer to the *XE500 BTOS System Capabilities Overview*.

Contents

Figures

Tables

Overview

As system administrator, you have access to the complete range of XEBTOS software capabilities.

This section is a list of the administrative functions, which are discussed in detail in other sections of this guide or in the other guide noted.

- installing system software (this is covered in the *XE500 BTOS Installation and Implementation Guide*)
- configuring system software to match your system's hardware configuration (this is covered in the *XE500 BTOS Installation and Implementation Guide*)
- initializing XE520 disk devices
- managing the XEBTOS file systems
- maintaining archive files of the data in the XEBTOS file systems
- managing user access to system files and resources
- managing applications that run in secondary partitions on XE520 processors
- controlling workstation cluster lines
- using the Command Line Interpreter (CLI)
- monitoring XE520 processor activity
- running the system in different operating modes
- interpreting and recovering from error conditions
- filling out User Communication Forms for reporting system problems to Unisys Customer Support Centers

Using Master Utilities and Master Commands

In addition to the local workstation BTOS utilities described in your *BTOS II Standard Software Operations Guide*, there are several utilities specific to the XE520 environment. These are master utilities, and you invoke most of them with master commands.

This section describes master utilities and master commands.

The administrative functions that use these commands are described in later sections of this guide. For a description of each master command function and its command form fields, refer to the *XE500 BTOS Operations Guide*, which also describes basic user operations that require these commands.

Because you use many of the master commands to administer the XE520 system, you should restrict user access to them. Section 6 describes the methods for controlling access to master commands.

The AdminAgent system service controls the execution of master utilities on XE520 processors, running one master utility at a time. Refer to the *XE500 BTOS Installation and Implementation Guide* for information on configuring the AdminAgent.

Commands and Utilities

The terms command and utility are sometimes used interchangeably, but a distinction exists between them. Command refers to the command name and command form used with the BTOS Executive mode. (For a complete description of the BTOS Executive, refer to your standard software operations documentation.) Utility refers to a program used internally by BTOS, usually as a result of a command.

Not all BTOS utilities or master utilities have command forms and are invoked by commands.

Running Master Utilities

For the most part, workstation utilities can freely access XE520 resources. You can use these utilities to maintain the XE520 as you would a workstation.

Master utilities, however, execute on XE520 processors and can access XE520 resources only. Most master utilities are special versions of workstation BTOS utilities; however, some master utilities do not have corresponding workstation utilities. These master utilities are used for resources specific to the XE520.

There are two ways to run master utilities:

- You can issue the corresponding command through the BTOS Executive. The Executive uses an XE520 AdminAgent to execute the command remotely at the XE520.

  You can invoke master utilities from the Executive only if they have a corresponding command form and only if the AdminAgent has been configured to use them.
- You can execute a utility's run file through the Command Line Interpreter (CLI) using Job Control Language (JCL). Refer to section 9 for more information about the CLI.

In all cases, master utilities execute on an XE520 processor. When used as a master utility parameter, a volume name always refers to an XE520 volume.

Using a Master Command

The following subsections describe procedures and conventions for using master commands. These commands are similar in operation to workstation BTOS commands.

Moving the Cursor within a Command Form

You can move through the fields of a command form by pressing RETURN, DOWN ARROW, and UP ARROW.

You move up through the fields by pressing UP ARROW.

When the cursor is in the last field of a form, you return the cursor to the first field by pressing RETURN or DOWN ARROW.

Optional Fields

Command forms can contain optional fields, enclosed in square brackets. You need not fill out an optional field to execute a function.

Optional fields that accept yes or no answers are configured with a default value. Other types of optional fields either are configured with default values or do not require an entry. The command descriptions in the *XE500 BTOS Operations Guide* discuss the optional fields and whether a field has a default value.

Executing a Master Command

To execute a master command, use the following procedure:

1 Enter the name of the master command at the BTOS Executive command line.

2 Choose one of the following:

- Proceed to step 4 to accept all system defaults or to activate a command that has no command form.
- Press RETURN to display the command form.

3 Fill in the appropriate fields of the command form.

4 Press GO.

The system executes the command.

Normally the command sends messages to the workstation screen to inform you of the command execution status.

Depending on the command you use, the system can generate a a report or listing and send it to the screen. These commands usually provide the option of sending the report or listing to a printable file so that you can have a printed copy of the information.

For more information about individual master commands, refer to subsequent sections of this guide and to the *XE500 BTOS Operations Guide.*

Initializing and Verifying XE520 Disks

This section provides information and procedures for:

- initializing disks to create new volumes
- reinitializing disks to correct volume fragmentation or add new disk bad spot information
- generating a listing of a volume's bad spots
- verifying a disk's hardware and volume structure integrity

Overview of XE520 Disks, Volumes, and File Systems

Each XE520 disk must be initialized (formatted) before you can store data on it. An XE520 disk can be a built–in disk or a Storage Module Device (SMD) disk.

Note: In this guide, the term built–in disk refers to a 5 1/4–inch hard disk that a File Processor (FP) controls. An XE520 can have an SMD disk built into its base enclosure, but this disk is controlled by a Disk Processors (DP) and is functionally different from other built–ins.

Once a disk is properly initialized to accept data, it is said to contain a volume. A disk is a hardware device; a volume is the complete file system unit of information stored on the disk. Each initialized disk in the system has a volume associated with it.

Because of this relationship, you can frequently substitute the device name of the disk drive for the volume name when referring to the information stored on the disk.

Disk Drive Device Names

XE520 disk devices are named according to their physical location in the system. The name conventions are different for disks controlled by FPs (built–in disks) and for disks controlled by DPs (SMDs).

Disks in the first enclosure controlled by an FP are named d0 through d3; those in a second enclosure are named d4 through d7; and so on.

SMD disks controlled by the first DP in the system (DP00) are named s0 through s5; SMDs controlled by the second DP in the system (DP01) are named s6 through s11; and so on.

Appendix B contains information on disk naming schemes.

Disk drive device names and passwords can be defined in:

- the FP's and DP's operating systems

  These names and passwords are assigned when a processor's operating system is generated. For more information about these parameters and operating system generation, refer to the *BTOS II Customizer Programming Guide*.
- the FP's and DP's configuration files

  For more information about these parameters in FP and DP configuration files, refer to the *XE500 BTOS Installation and Implementation Guide*.

Volume Names

Volume naming conventions for XE520 volumes are the same as for workstation volumes.

You assign a BTOS volume name to a disk when you initialize it with the mIVolume utility. Follow these rules when selecting a volume name:

- Use a maximum of twelve characters, including any alphanumeric characters, hyphens, and periods.
- Do not duplicate any other BTOS volume or device name.
- Do not use any of these as a volume name: d0, d1, d2, d3, and so on; s0, s1, s2, s3, and so on; Nul; or Kb.
- Do not use a volume name that begins with: Comm, Lpt, Spl, Tape, Vid, or QIC.

The XEBTOS File System and File Names

In canonical form, the BTOS file system has a four-level structure:

- Node. A node is the name of a workstation or other point within a communications network.
- Volume. A volume is associated with each initialized disk in the system.

- Directory. Each volume can contain one or more directories. All directories are at the same level (that is, a directory cannot contain another directory).
- File. Each directory can contain one or more files.

A full BTOS file name has the following format:

{node}[volname]<dirname>filename

node is the name of the point in the communications network where the file is stored. Specifying the node is optional; when not specified, it is assumed to be the local workstation or terminal.

volname is the name of the volume on which the file is stored. You can normally use the disk name associated with the volume instead of the actual name of the volume.

If a workstation and an XE520 disk device or volume both have the same disk name or volume name, you precede the XE520 name with an exclamation mark (!).

dirname is the name of the directory in which the file is stored.

filename is the name assigned to the file.

Refer to your *BTOS II Standard Software Operations Guide* for detailed information about file name specifications.

Overview of Volume Initialization

The mIVolume (Initialize Volume) utility prepares an XE520 disk for use as a system volume. The mIVolume utility:

- formats the disk surfaces into 512-byte sectors
- performs surface tests to identify defects
- writes volume control structures (Volume Home Blocks, Master File Directory, Allocation Bit Map, and File Header Blocks) onto the disk Volume Home Block Master file directory
- creates system files

Every disk must be initialized before use.

Built-in disks and MD3 disks are normally initialized before they are shipped to customers or by the field representative when the system is installed.

You need to reinitialize disks under certain conditions, such as:

- when system performance is degraded due to volume fragmentation (refer to Volume Fragmentation)
- when system files or volume control structures are unreliable due to disk bad spots
- to change the size of certain volume-based files for the designated system disk, such as the system image file, the log file, and the system crash file
- to change the maximum number of directories or files allowed on the volume
- to set the volume control structures to use primary headers only (as opposed to having primary and secondary headers)
- to set protection for the <sys> directory
- to enter new disk bad spots

All disks come with a listing of known bad spots, or defective areas on the disk surface. When you initialize a disk for the first time, the system enters this list into a bad spot file, called badblk.sys, in the volume's <sys> directory. The volume uses the bad spot list to track the areas where data cannot be stored.

The badblk.sys file is a coded file. The mIVolume and Bad Spot Report utilities generate a text listing of the bad spots.

Volume Fragmentation

BTOS volumes can become fragmented over time. When you create or extend files, BTOS tries to allocate a single disk extent to store the new data. (A disk extent is one or more contiguous disk sectors that make up all or part of a file.)

If you have recently initialized a volume, the system can easily find a single disk extent that is large enough to satisfy your request. However, if you have created and deleted the files many times since volume initialization, the disk extents available for allocation can be scattered. In

this case, the volume is said to be fragmented, because the system must allocate two or more smaller disk extents that have a total size sufficient to fulfill the request.

If a volume becomes fragmented, performance is affected several ways:

- The system requires more time to create or extend a file because it must access more sectors of the Bit Allocation Map to find enough disk extents to satisfy the request.
- The system requires more time to process a file sequentially because disk sectors that are logically consecutive may not be physically consecutive.
- Fewer files can be open concurrently because each open file requires allocation of a File Area Block in memory for each disk extent of that file.

You can correct this condition in two ways:

- You can back up the volume to archive tapes or disks and reinitialize the disk. When the archived data is restored, files are placed on contiguous disk extents. (For information about backing up and restoring files, refer to section 6.)
- You can consolidate multiple free areas on a disk using the mDisk Squash utility.

  This utility does not consolidate a multiple–extent file into a single–extent file (as the backup, reinitialize, and restore operation does) but helps reduce the fragmentation of the free areas on a disk. Refer to Reducing Fragmentation with the mDisk Squash Utility.

Initializing XE520 Volumes

To format XE520 disks, you use the mIVolume utility. You also use it to:

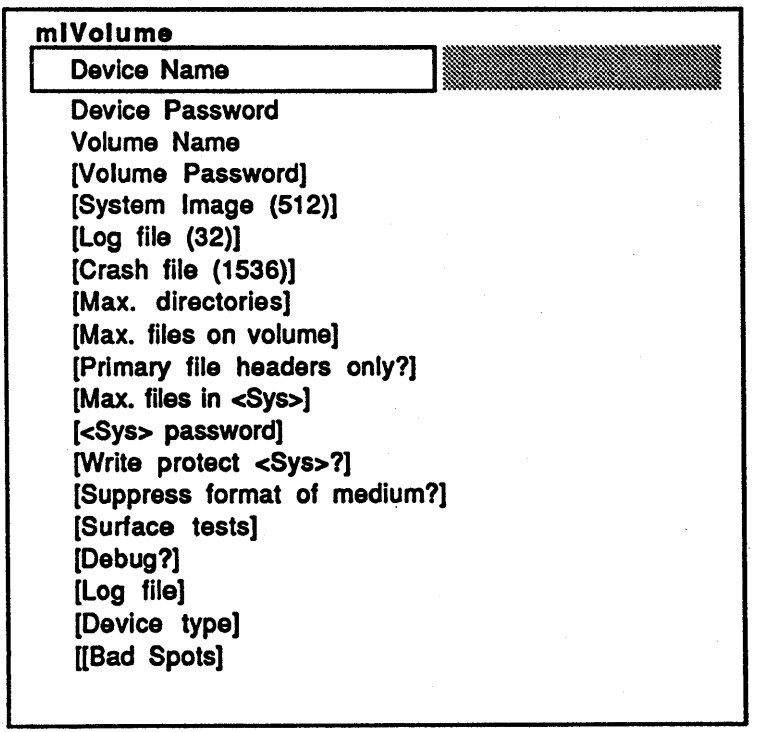
- assign the volume name
- assign the volume password
- determine the maximum number of directories and files that can be created on the volume
- determine whether primary and secondary file headers, or just primary file headers are to be used

- assign a password to the <sys> directory
- determine whether the <sys> directory is to be write-protected
- add to the volume's bad spot listing

To activate the mIVolume utility, you enter **MIVOLUME** at the Executive command line and press RETURN.

The system displays the MIVOLUME command form shown in figure 3-1.

Figure 3-1 **MIVOLUME Command Form**



You must enter parameters in three MIVOLUME fields:

- In the **Device name** field, enter the name of the device that contains the volume you want to format.
- In the **Device password** field, enter the password for the device. Unless a user modifies it, the device password is the same as the device name.
- In the **Volume name** field, enter a name (a maximum of 12 characters) to identify the volume. This name must not duplicate any device or active volume name.

The MIVOLUME command form has 16 optional fields (enclosed in square brackets). The defaults for these fields are set for a system volume. Refer to table 3–1 for information about each optional field.

When you complete the MIVOLUME command form, you press GO. The initialization of the volume begins. You must respond to the prompts that appear during the initialization procedure.

The mIVolume utility verifies the consistency of the parameters specified in the command form, opens a log file for the initialization operation (if a log file is specified), and displays the values chosen for the sizes of the volume control structures.

Table 3–1 **MIVOLUME Command Form Fields**

Field	Action/Explanation
Device Name	Enter the physical device name of the disk drive to be initialized. Default names for XE520 built–in disk drives are d0, d1, d2, and so on. Default names for SMD disk drives are s0, s1, s2, and so on.
Device Password	Enter the password (if one exists) for the drive device. To assure system security, the password does not display as you enter it. The default password for a disk device is the same as its default device name.

Table 3–1 **MIVOLUME Command Form Fields** (continued)

Field	Action/Explanation
Volume Name	Enter a name, with up to 12 characters, to be assigned to the volume. It must not contain [,] , <, >, {, or }, must not begin with a device name or Comm, CTOS, Kbd, Lpt, QIC, Spl, Sys, Tape, or Vid, and must not duplicate another volume name.
[Volume Password]	If you do not specify a password, the volume is unprotected. No password is necessary to access it, and none of its directories or files can have passwords. To assign a password to this volume, enter a maximum of 12 characters. Depending on the file protection level assigned, users use this password when they create files or directories, or when they open files on this volume. You can assign or modify a volume's password later with the MCHANGE VOLUME NAME command.
[System Image (def = 512)]	This is the number of sectors required for a system image (that is, the master operating system run file). For the XE520 system disk, the system image size should be at least 512 sectors. To initialize a nonsystem disk, enter 0. mIVolume creates an empty file, [Sys]<Sys>SysImage.sys, whose size is the specified number of sectors. You cannot install or copy the SysImage.sys file to an initialized volume without using mIVolume to allocate space for it. If the area for SysImage.sys is smaller than the SysImage.sys file being copied, the system displays an error message. You must reinitialize the disk to recover.

Table 3–1 **MIVOLUME Command Form Fields** (continued)

Field	Action/Explanation
[Log file (32)]	Enter the number of sectors required for the system log file, [sys]<sys>log.sys. The default value is 32. The system log, displayed through the PLOG command, reports system status and error conditions. Specify a larger number if the log file tends to fill before it is convenient for you to print it. Log entries are written only to system disks. If this volume is not to contain the master operating system, enter 0.
[Crash file (1536)]	Enter the number of sectors required for the system crash file. The system crash file stores a dump of the master processor's memory image when the system starts up or crashes. Allocate two sectors for each 1 kB of processor memory to be dumped. For non–production systems, the recommended value is 1536 sectors.
[Max. directories]	Leaving this field blank directs the system to set the maximum number of directories for this volume according to the volume size. To specify your estimate of the maximum number of directories to be created, enter that number, which must be less than 65535. You can reallocate a volume structure whose size is derived from this parameter only by reinitializing the volume.
[Max. files on volume]	Leaving this field blank directs the system to set the maximum number of files for this volume according to the volume size. To specify your estimate of the maximum number of files, enter the number. For a 50–Mb disk, 3000 is the recommended value. You can increase this number later only by reinitializing the volume.

Table 3–1 **MIVOLUME Command Form Fields** (continued)

Field	Action/Explanation
[Primary file headers only?]	The default is no. Leave this field blank to direct the system to allocate space for both primary and secondary File Header Blocks. Enter yes to allocate space for a primary File Header Block only. This destroys the secondary file structure, increasing the chance of file loss through data errors but greatly increasing the amount of disk space.
[Max. files in <Sys>]	Leaving this field blank directs the system to set the maximum number of files in the <Sys> directory according to the volume size. To specify a maximum number of files other than the default, enter the number. For a volume to be used for data or archive, 15 is the recommended value. For a volume to be used for booting the operating system, add 25 percent to the number of files that will go in the <Sys> directory. File system performance degrades if the directory exceeds 80 percent of capacity. You can change the size of <Sys> only by reinitializing the volume.
[<Sys> password]	Enter a password that you want to assign to the <sys> directory, up to a maximum of 12 characters. If you enter a password, it is valid only if you also specify a volume password. The default is no password, which leaves the files unprotected. If you enter a password, users need it to create or open files in the <sys> directory (depending on the protection level assigned to each file).
[Write protect <Sys>?]	Enter yes to set the default file protection level of the files in the <sys> directory to modify–protected (protection level 5). The default is no, which sets the protection level to unprotected (protection level 15). Protection is valid only if you have also entered passwords for the volume and <sys> directory.

Table 3–1 **MIVOLUME Command Form Fields** (continued)

Field	Action/Explanation
[Suppress format of medium?]	Enter yes to suppress formatting. Suppressing the formatting of the disk reduces the time to reinitialize a previously initialized valid disk. The utility produces an error message when writing system volumes. The default is no. When initializing a new disk or reinitializing a corrupted disk, leave this field blank.
[Surface tests]	Enter the number of surface tests to be performed on the disk. A surface test writes data to and then reads the data from each sector of the disk to check for bad spots. However, this is not a substitute for entering known bad spots in a [Bad Spots] field. The default is 1 if the disk contains a valid volume, 8 if it does not. The recommended number is 3 for built–in disks, 8 for SMDs. A surface test for a built–in disk takes approximately 20 to 30 minutes; for an SMD, it takes approximately 20 to 45 minutes. To shorten the time required to reinitialize a noncorrupted disk, enter 0.
[Debug?]	The default is no. If you specify yes, mlVolume generates information of interest to Unisys engineers and takes more time to run.
[Log file]	To create a print file of the mlVolume status report, enter a file name. If the file already exists, the new report is appended to it. You can view the file at your screen or print it. To send the report to a printer automatically and not save the print file, enter a print queue name, enclosed in square brackets. The default is Vid; the report displays on the screen.

Table 3–1 **MIVOLUME Command Form Fields** (continued)

Field	Action/Explanation
[Device type]	Enter the device type of the disk you are initializing. The default is the type in the volume structures. .cnf file names have the form lvxxx.cnf, where xxx is the device type. Examples of valid type names are: TOSHIBA85 for a 72.2 MB built–in disk. MEMOREX166 for an SMD disk.
[Bad Spots]	For information about entering bad spots, refer to Bad Spots.

To avoid inadvertently destroying a volume that contains valid data, mIVolume reads the sectors where a BTOS Volume Home Block (VHB) exists. If mIVolume locates a VHB, the information contained in it displays. This information includes the volume name, the date it was created, the date it was last modified, and the number of free pages and file headers. It also prompts for confirmation to reinitialize the volume.

Caution: Reinitializing a valid volume causes all data stored on the volume to be lost. Before reinitializing a valid volume, always back up any files that should be saved.

If you confirm the reinitialization and the old volume has a password, the system prompts you to supply the password. mIVolume echoes a pound sign (#) for each password character you enter. You press RETURN to terminate the password.

As the first step of initialization, mIVOLUME formats the disk and performs the number of surface tests specified in the command form. Each bad spot is recorded in the bad sector file ([sys]<sys>badblk.sys). When all surface tests are completed, the list of bad sectors displays. This list includes previously known bad spots, any bad spots detected during the surface tests, and bad spots entered in the command form.

After listing the bad spots, mIVolume writes the volume control structures on the disk. The volume control structures include: the files Mfd.sys, FileHeaders.sys, BadBlk.sys, and DiagTest.sys; the Volume Home Block; and the Allocation Bit Map.

Note: The DiagTest.sys file is reserved for diagnostics. You cannot delete it and should not use it to store data. It is not backed up during a backup operation.

The DiagTest.sys file is allocated on the second–to–last track of each XE520 disk.

Finally, mIVolume creates the system files SysImage.sys and CrashDump.sys. These two files are needed if the volume is to contain a master operating system (that is, the XE520 could boot from the volume). However, nothing is written to these two files at this time.

Guidelines for Initializing a Volume

The following is a list of guidelines for using the mIVolume utility:

- mIVolume does not allow you to initialize a volume currently in use.
- If you are initializing a disk for the first time, you must enter the bad spots listed in the manufacturer's report that accompanies each disk.

  For more information on how to enter bad spots, refer to Entering Bad Spots.
- Once you initialize a volume, you use the BAD SPOT REPORT command to create a record of bad spots. You should save this record; you must have it if the volume becomes corrupted.

  You can use the information the BAD SPOT REPORT command generates to create a file whose name you can enter in a **[Bad spot]** field of the MIVOLUME command form. You then need not enter each bad spot individually. Refer to Entering Bad Spots.
- If you try to initialize a corrupted disk (that is, a disk the system cannot mount), you must enter all known bad spots.
- If you are initializing a volume for the first time or a volume that is corrupted, you must specify a device type in the MIVOLUME command form.

- If you are reinitializing a valid volume, you do not have to specify the device type.
- If you interrupt the initialization operation, the volume becomes invalid and you must reinitialize the disk as a new disk.
- If you are reinitializing a valid BTOS volume, you can suppress the medium formatting operation by specifying **yes** in the **[Suppress format of medium?]** field. This reduces the time required to initialize the disk.
- The value specified for the maximum number of directories is rounded up to the nearest multiple of 15 to produce the size of the file [sys]<sys>Mfd.sys. Since you can expand this file only by reinitializing the disk, it is important to determine an appropriate value.
- mIVolume allocates File Header Block (FHB) sectors for each file. The value specified for the maximum number of files is multiplied by 1.5 to allow for a certain number of extension FHBs. This product is then rounded up to an even multiple of three times the number of sectors on each cylinder of the disk.

  mIVolume locates these FHBs on the disk to ensure that the primary and secondary FHBs are located on different cylinders, at different rotational positions, and (for system volumes) on different disk surfaces.

  Since you can increase this number only by reinitializing the disk, it is important to determine an appropriate value.
- The value specified for the maximum number of files in the <sys> directory is rounded up to the nearest multiple of 14. Because system performance is degraded if the volume is a system volume and this directory is more than 80% full, you must allow for more files than you needed.

  Since you can expand this number only by reinitializing the disk, it is important to determine an appropriate value.

Bad Spots

If you are initializing a disk for the first time or are initializing a corrupted disk (that is, a disk no longer mountable by the system), you must enter a listing of known bad spots in the **[Bad spots]** field of the MIVOLUME command form.

Once you initialize a disk, the system maintains a coded list of the bad sectors in the file <sys>badblk.sys.

A sector is identified as defective (and therefore omitted from the Allocation Bit Map) for three reasons:

- You specified a bad spot in the sector in the MIVOLUME command form.
- Before initialization, the disk contained a valid BTOS volume and the sector was previously identified in the bad sector file as defective.
- The sector failed the surface tests of the mIVolume utility.

Identifying Bad Spots

You identify bad spots by their location on the disk. A disk location is designated by cylinder/head/sector or cylinder/head/byte (a cylinder is sometimes referred to as a track).

Bad spot locations are designated by the format:

c/h/#s or c/h/b/l

c is the cylinder number
h is the head number
s is the sector number
b is the byte number

l is the length of the defect. You should always use the value 1 to specify the length of the defect.

All numbers used in identifying bad spots are in decimal.

The mIVolume utility converts all bad spot designations in the byte format to the sector format. When specifying bad spots in the MIVOLUME command form, you should use the sector format rather than the byte format.

Listing Current Bad Spots

To get a listing of bad spots, you specify a log file or print file name in the MIVOLUME or BAD SPOT REPORT command form when you use one of these utilities. (Refer to Executing the BAD SPOT REPORT Command.)

The report these utilities generate includes a listing of the bad spots currently stored in the <sys>BadBlk.sys file. You can use this listing to create a file whose name you can enter in the **[Bad spot]** field instead of entering each bad spot individually. For more information, refer to Entering Bad Spots.

Entering Bad Spots

To enter bad spots in the MIVOLUME command form, you can enter each bad spot individually or enter the name of a file that contains a list of the bad spots.

To enter bad spots individually, you type each bad spot designation in one of the form's three **[Bad spots]** fields. Do not break an individual bad spot entry over two fields.

If all the entries do not fit in the three fields, you must use the BTOS Editor to create a file that lists the bad spots, separating the bad spot designations by a space or carriage return. Then, in one **[Bad spots]** field, enter **@BadSpotFile,** where BadSpotFile is the file you have listed the bad spots in.

If you saved the print file generated when you executed the MIVOLUME or BAD SPOT REPORT command on the volume, you can edit this file.

To create a bad spot file, use the following procedure:

1 Copy the print file to the file you enter in a **[Bad spot]** field.

2 Use the BTOS Editor to delete all nonpertinent information from the print file.

The only information to keep in this file is the bad spot listing itself.

3 In the **[Bad spot]** field, enter **@BadSpotFile**, where BadSpotFile is the name of the file you edited.

Note: If you are executing mlVolume through a CLI run statement and are listing bad spots individually, you must enclose the list of bad spot entries in parentheses.

Note About Bad Spots

Built-in disk drives are formatted for 591 bytes per sector (512 of which are for data), 16 sectors (0–15), 7 heads (0–6), and 645 cylinders (0–644). You cannot enter a defect on a cylinder greater than 644 or having a byte value greater than 9455. Also, you do not enter bad spots for cylinder 644, head 6; it is the last track on the disk and is reserved. You should ignore bad spot information in the form degree n.

Executing the BAD SPOT REPORT Command

The BAD SPOT REPORT command invokes a utility that tests XE520 built-in and SMD disks for soft and hard errors without destroying valid data.

To use the Bad Spot Report utility, you enter BAD SPOT REPORT at the Executive command line. The system displays the BAD SPOT REPORT command form shown in figure 3–2.

You must enter a parameter in the **Volume or device name** field. Refer to table 3–2 for information about the three optional fields.

Figure 3–2 **BAD SPOT REPORT Command Form**

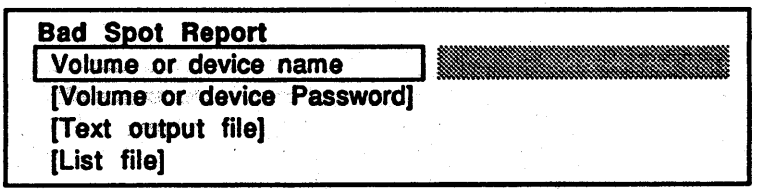


Table 3–2 **BAD SPOT REPORT Command Form Fields**

Field	Action/Explanation
Volume or device name	Enter the name of the volume or device you want to check for bad spots. The default is the current working volume.
[Volume or device password]	Enter the password, if any, of the volume or device to be checked. The default is no password.
[Text output file]	Enter the name of a file to hold the list of bad spots, in the format the **[Bad Spots]** fields of the MIVOLUME command require. The default is no file.
[List file]	Enter the name of a file to hold bad spot information in the form of a table, showing each bad block's cylinder, head, sector, and size. If the file does not exist, it is created; if it already exists, the new information is appended to it. The default is screen display only.

Reducing Fragmentation with the mDisk Squash Utility

To reduce volume fragmentation, you can execute the mDisk Squash utility on an XE520 disk. This utility consolidates multiple free areas on a disk into a single disk–extent free area. The analysis option allows you to determine the current state of a disk.

This utility does not consolidate all multiple–extent files into single–extent files, only free areas. To consolidate all multiple–extent files into single–extent files, you must back up, reinitialize, and restore the disk. Refer to section 6 for instructions on backing up and restoring a disk.

Executing the MDISK SQUASH Command

Note: You cannot execute the mDisk Squash utility on any disk that has open files. If you attempt to execute the utility on a disk with open files, it aborts and returns an appropriate message.

Therefore, before squashing a disk, you must make sure that no one is currently using it and that no files are open. Because the Command Line Interpreter (CLI) always keeps an open file on the system disk, you must perform the procedure outlined in Preparing the System Disk for a Disk Squash before attempting to squash it.

To execute the MDISK SQUASH command on a disk other than the system disk, use the following procedure:

1 Enter **MDISK SQUASH** at the Executive command line.

2 Press RETURN.

The system displays the MDISK SQUASH command form shown in figure 3–3. Table 3–3 lists the fields in the MDISK SQUASH command.

3 Enter the device or volume name of the disk you want to squash in the **Volume or device name** field.

4 Enter the volume's password, if one exists, in the **[Volume password]** field.

5 Choose one of the following:

- Leave the **[Analyze only?]** field blank to squash the disk and receive the analysis report.
- Enter yes in the **[Analyze only?]** field to receive an analysis report only and not squash the disk.

6 Choose one of the following options:

- Enter a file name in the **[Print file]** field to obtain a file of the report.
- Enter a print queue name enclosed in square brackets to print out the report automatically without creating a print file.

  Note: If your system has only one disk, you cannot use this option.

7 Press GO when you have filled in the appropriate fields of the command form.

Figure 3–3 **MDISK SQUASH Command Form**

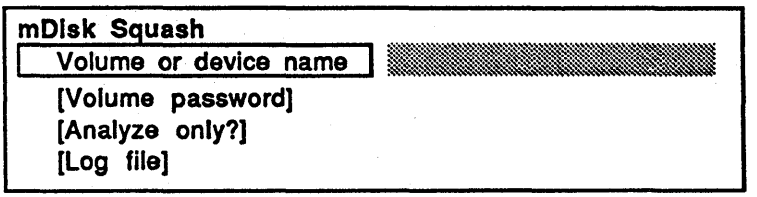


Table 3–3 **MDISK SQUASH Command Form Fields**

Field	Action/Explanation
Volume or device name	Enter the name of the volume or of the disk device to be squashed.
[Volume password]	Enter the volume password, if one exists.
[Analyze only?]	To squash the disk and receive the analysis report, leave this field blank. To receive an analysis report only and not squash the disk, enter yes.
[Log file]	Enter the name of the file you want the report written to. Specify a print queue, enclosed in square brackets, to have the listing copied to a temporary print file and sent to the specified print queue. If you do not name a print file or print queue, the system sends the information to the screen. If you have only one disk in your system, you cannot use this option.

The utility first displays general information about the volume, including the volume name, when it was created, the last time data on it was modified, the number of free file headers, the number of files, and the size in sectors (512 bytes). The utility then analyzes the disk and returns the following information:

- the number of used sectors
- the number of free sectors
- the number of free areas
- the maximum free area (the size of the largest free area)
- the average size of all the free areas
- the number of bad spots
- the user count (the number of open files on the volume)

  If any files are open, the utility terminates. You must make sure that no one is using any files on the volume before you attempt to squash it.

After analyzing the disk, the utility performs the squash operation and displays an updated version of the previously listed information. Normally, the number of free areas decreases, the size of the largest free area increases, and the average size of a free area increases.

A sample report is shown in figure 3–4.

Figure 3–4 **Sample mDisk Squash Report**

```
Oct 19, 1988  10:44  AM

Volume Name                    :   SMDO
Creation Date/Time             :   Thu Apr 9, 1987 12:58 AM
Last Modification Date/Time    :   Sun Apr 12, 1987 9:04 AM
Number of Free File Headers    :   22111
Number of files                :   7888
Total number of sectors        :   263360

Analyzing the disk


Used sectors                   :   96007
Free sectors                   :   167353
Free Areas                     :   7910
Maximum free area              :   27512
Average size of free area      :   21
Number of bad spots            :   45
User count                     :   0

Disk Squash is in progress, Do NOT interrupt


Disk squash completed


Analyzing the disk

Used sectors                   :   96007
Free sectors                   :   167353
Free Areas                     :   98
Maximum free area              :   27512
Average size of free area      :   1707
Number of bad spots            :   45
User count                     :   0
```

Managing XE520 File Systems

This section provides procedures and information for managing XE520 volumes and file systems.

File system management includes:

- assigning volume names and passwords
- obtaining volume status information
- creating and removing directories

Assigning Volume Names and Passwords

You assign a volume name and password by using the MIVOLUME or the MCHANGE VOLUME NAME command.

Note: To change the BTOS system disk volume name and password, you must use a special procedure. Refer to Changing System Disk Volume Name and Password.

You can use mIVolume if you are initializing a disk for the first time, have already backed up valid data on the volume, or do not intend to save the data on the volume. Refer to the procedure for initializing volumes in section 3.

Note: You cannot use the MIVOLUME command to change passwords because the volume password entered in the **[Volume password]** field must be the currently assigned password. You must use the MCHANGE VOLUME NAME command to change a volume password.

If you want to change the volume name and/or password without initializing the volume, you use the MCHANGE VOLUME NAME command.

The command form for the MCHANGE VOLUME NAME command is shown in figure 4–1; its fields are explained in table 4–1.

Figure 4–1 **MCHANGE VOLUME NAME Command Form**

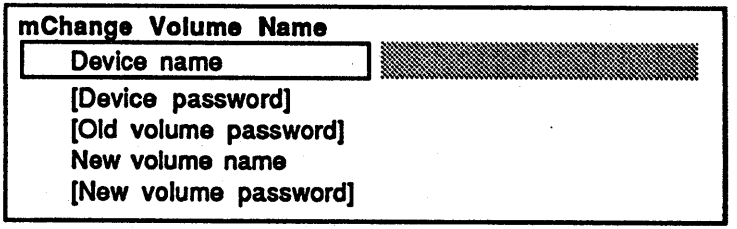


Table 4–1 **MCHANGE VOLUME NAME Command Form Fields**

Field	Action/Explanation
Device name	Enter the device name of the disk drive whose volume name you want to change. Default names for XE520 built–in disk drives are d0, d1, d2, and so on. Default names for SMD disk drives are s0, s1, s2, and so on.
[Device Password]	Enter the password for the drive device, if desired. A pound sign (#) displays for each password character entered, to ensure system security. The default password for a disk device is the same as its default device name. The default is no password used.
[Old volume password]	Enter the current volume password. If no password is currently assigned, leave this field blank. If you want to change only the volume name, the old volume password must still be specified. The default is no password.

Table 4–1 **MCHANGE VOLUME NAME Command Form Fields** (continued)

Field	Action/Explanation
New volume name	Enter the new name to be assigned to the volume. It can have up to 12 characters and must not duplicate another device or volume name. If you are changing only the password, enter the current volume name.
[New volume password]	Enter the new password to be assigned to the volume, using a maximum of 12 characters. To leave the current password unchanged when the mChange Volume Password utility runs, leave this field blank. To remove the current volume password and leave the volume unprotected, enter two single quotes with no space between them (''). The default is no new password.

To change a volume name and/or password, use the following procedure:

1 Enter **MCHANGE VOLUME NAME** at the Executive command line.

2 Press RETURN.

3 Enter the volume's disk device name in the **Device name** field of the MCHANGE VOLUME NAME command form.

4 Enter the volume disk device password in the **Device password** field.

5 Enter the current volume password in the **[Old volume password]** field.

If the volume does not currently have a password, leave this field blank.

6 Enter the new volume name in the **New volume name** field.

Enter the current volume name if you are executing the command to change the volume password and not the name.

7 Enter the new volume password in the **[New volume password]** field.

Leave this field blank if you are executing the command to change the volume name and not the password.

To remove the current password and have no password assigned to the volume, enter two single quotes with no space between them ('').

8 Press GO.

You have now changed the volume's name and/or password as you specified in the command form.

Note: To set file security for files on a volume, you must assign a password to the volume.

Files on a volume with no assigned password are unprotected, regardless of whether the volume's directories or files have passwords, or what file protection levels have been set.

For more information about volume passwords and system security, refer to section 5.

Changing System Disk Volume Name and Password

Because the system disk is used when the mChange Volume Name utility is running, you cannot use the MCHANGE VOLUME NAME command to change the volume name or password of the BTOS system disk.

To change the system disk volume name or password, use the following procedure:

1 Reboot the system from tape, with the keyswitch in the MANUAL position.

The booting system menu appears.

2 Enter **3** as the option number, and press RETURN.

3 Enter each of the following as the system prompts you for it. Press RETURN after each.

- device name
- device password
- old volume password
- volume name
- new volume password

To leave an item unchanged, enter the current name or password. For example, to change only the volume name, you enter the following:

- current device name
- current device password
- current volume password
- desired new volume name
- current volume password (again)

Getting Volume Status Information

To get status information about a volume, use the VOLUME STATUS command. When the command executes, the following information about the volume displays:

- Volume name
- Date created
- Date last modified
- Number of free sectors. This information indicates the amount of free memory on the volume (a sector is 512 bytes).
- Number of free file headers. This information indicates the files that you can still create on the volume.
- A listing of the volume's directories

The command form for the VOLUME STATUS command is shown in figure 4–2, and its fields are explained in table 4–2.

To execute the VOLUME STATUS command on the current working volume and accept all default parameter values, you enter **VOLUME STATUS** at the Executive command line and press GO. The volume status report displays on the screen.

To execute the VOLUME STATUS command on a volume other than the current working volume, you enter **VOLUME STATUS** at the Executive command line and press RETURN. When you fill in the command form's fields and press GO, the volume status report displays on your screen.

A sample volume status report is shown in figure 4–3.

Figure 4–2 **VOLUME STATUS Command Form**

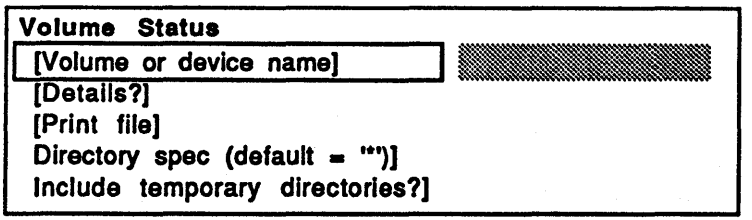


Table 4–2 **VOLUME STATUS Command Form Fields**

Field.	Action/Explanation
[Volume or device name]	Enter the name of the volume or the disk the volume is mounted on. If this field is left blank, the system returns status information on the current working volume.
[Details?]	The default is no; the system displays status information with no details. Enter y for detailed status information.
[Print file]	You can enter the name of a file for storing the status information, and from which you can print that information. Or, you can enter an appropriate device name, such as [Lpt], to print the information directly. The default is to display the information on screen only.

Table 4–2 **VOLUME STATUS Command Form Fields** (continued)

Field.	Action/Explanation
[Directory spec (default = '*')]	To display status information for specific directories only, enter the name of each directory, enclosed in angle brackets (for example, <office>). The default is to display information for all directories in the volume.
[Include temporary directories?]	To include temporary directories, enter y. The default is no.

Figure 4–3 **Sample Volume Status Report**

```
Status of volume sys

Created                          Aug 15, 1988  2:28 PM
Last modified                    Sep 19, 1988  12:18 AM
Number of free sectors           58316
Number of free file headers      2458

Directory Name      Sectors      Default Protection
Admin                  3              15
Exec                   3              15
$00001                 6              15
Backup                 3              15
SYS                   67              15
Intercom1500           3              15
SWP1                   3              15
SWP2                   3              15
$001                   3              15
SWP3                   3              15
$002                   3              15
SWP4                   3              15
OSBuild                6              15
attdocs                3              15
$003                   3              15
WP                     7              15
cmd                    3              15
Sp1                    3              15
```

Creating Directories

To create a directory on an XE520 volume, you use the MCREATE DIRECTORY command. This command optionally allows you to:

- set the default protection level for any files created in the directory

  Valid protection levels you can enter in the command form are 15 (unprotected; the default), 5 (modify protected), and 0 (access protected). For more information about file protection levels, refer to section 5.
- set the maximum number of files that the directory can have
- set a password for the directory

The command form for the MCREATE DIRECTORY command is shown in figure 4–4, and table 4–3 explains its fields.

Figure 4–4 **MCREATE DIRECTORY Command Form**

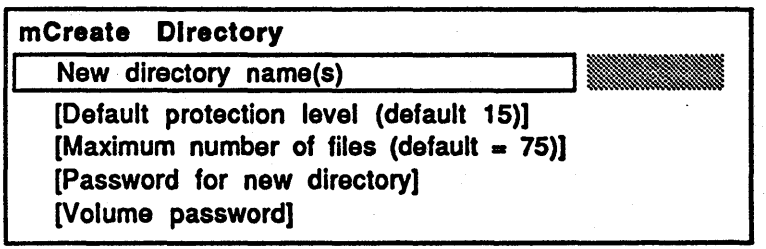


Table 4–3 MCREATE DIRECTORY Command Form Fields

Field	Action/Explanation
New directory name(s)	Enter the name of the directory you want to create, using a maximum of 12 characters for each directory name. Do not use the reserved characters {, }, [,], <, and >. If you are not creating the directory on the current working volume, include the volume name in square brackets and enclose the directory name in angle brackets (for example, [xyz]<abc>).
[Default protection level (default = 15)]	Enter the protection level you are assigning to all files created in the directory. The valid protection levels are 15 (unprotected), 5 (modify–protected), and 0 (access–protected).
[Maximum number of files (default = 75)]	Enter the maximum number of files you want the directory to handle. The maximum number of files you assign in this field is divided by 25 to obtain the number of sectors allocated to that directory. The assumption is that if you have 25 names, they contain 17 characters each. If the names are longer, fewer than the assigned number will actually fit.
[Password for new directory]	Enter the password you want to assign to the new directory, using a maximum of 12 characters. Leave this field blank if you do not want the directory to have a password. Files in the directory cannot be password–protected unless the directory has a password. The default is no password.
[Volume password]	Enter the password for the volume you want the directory created on. If the volume has no password, leave this field blank.

To execute the MCREATE DIRECTORY command, use the following procedure:

1 Enter **MCREATE DIRECTORY** at the Executive command line.

2 Press RETURN.

The command form shown in figure 4–4 displays. Table 4–3 explains the command form fields.

3 Choose one of the following:

- Enter the full directory specification (for example, [xyz]<abc>) in the **New directory name(s)** field if the directory is to be in a volume other than the current working volume.
- Enter the directory name alone in the **New directory name(s)** field if the directory is to be in the current working volume.

4 Enter the desired protection level in the **[Default protection level (default = 15)]** field to use a protection level other than the default level for files in the directory.

5 Enter the desired number in the **[Maximum number of files (default = 75)]** field to specify a maximum number of files for the directory other than the default.

6 Enter a password in the **[Password for new directory]** field to assign a password to the directory.

You cannot password–protect files in the directory unless you assign a directory password.

7 Enter the password of the volume you are creating the directory on, if one exists.

8 Press GO when you have filled in the appropriate fields.

You have created the directory as you specified in the command form.

Removing Directories

To remove a directory on an XE520 volume, you use the MREMOVE DIRECTORY command. This command optionally allows you to:

- delete all files in the directory automatically before the directory is removed
- confirm the deletion of each file

To execute the MREMOVE DIRECTORY command, use the following procedure:

1 Enter **MREMOVE DIRECTORY** at the Executive command line.

2 Press RETURN.

The command form shown in figure 4-5 displays. The command form fields are explained in table 4–4.

3 Choose one of the following:

- Enter the full directory specification (for example, [xyz]<abc>) in the **Old directory name(s)** field if the directory is in a volume other than the current working volume.
- Enter the directory name alone in the **Old directory name(s)** field if the directory is in the current working volume.

4 Enter the password in the **[Volume or directory password]** field if the directory or its volume has a password.

5 Enter yes in the **[Delete all files in directory?]** field if you want all files in the directory deleted automatically before the directory is removed.

You cannot remove the directory unless there are no files in it.

6 If you entered yes in the **[Delete all files in directory?]** field and want the utility to prompt you to confirm the deletion of each file, enter yes in the **[Confirm each while deleting?]** field.

7 Press GO.

You have removed the directory.

Figure 4–5 **MREMOVE DIRECTORY Command Form**

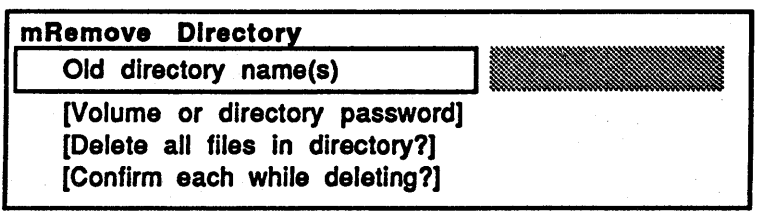


Table 4–4 **MREMOVE DIRECTORY Command Form Fields**

Field	Action/Explanation
Old directory name(s)	Enter the name of the directory you want to remove. If it is not on the current working volume, you must include the volume name in the directory specification (for example, [xyz]<abc>).
[Volume or directory password]	If the directory or its volume has a password, enter the password here. The default is no.
[Delete all files in directory?]	Enter yes if you want all files in the directory deleted before you remove the directory. If you leave this field blank and there are files in the directory, the utility does not remove the directory. A directory must be empty of all files before you can remove it. The default is no.
[Confirm each while deleting?]	This field is applicable only if you entered yes in the [Delete all files in directory?] field. Enter yes to have the utility prompt you to confirm the deletion of each file in the directory. Enter no or leave the field blank to have the utility delete all files in the directory automatically. The default is no.

Setting Directory Security

You can use the MSET DIRECTORY PROTECTION command to change the default file protection levels for files in a directory and to add, change, or remove a directory's password.

To execute the MSET DIRECTORY PROTECTION command, you type **MSET DIRECTORY PROTECTION** at the Executive command line and press RETURN. The system displays the MSET DIRECTORY PROTECTION command form shown in figure 4–6.

The MSET DIRECTORY PROTECTION command form has two required and two optional fields. You can leave the optional fields blank to accept the defaults or enter parameters to override the defaults. Refer to table 4–5 for information about each field.

When you complete the MSET DIRECTORY PROTECTION command form, you press GO. The system sets the new default file protection level or changes the directory password as you specified in the command form.

Figure 4–6 **MSET DIRECTORY PROTECTION Command Form**

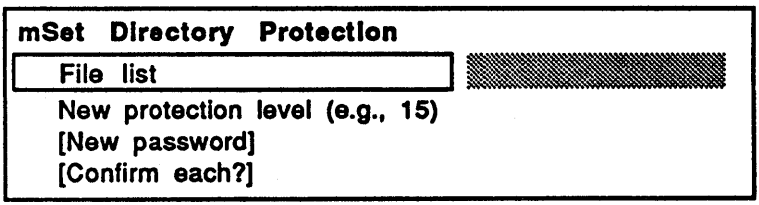


Table 4–5 **MSET DIRECTORY PROTECTION Command Fields**

Field	Action/Explanation
File list	Enter the names of the files to be protected.
New protection level (e.g., 15)	Enter the protection level to be assigned to the files. To keep the current default protection level, leave this field blank.
[New password]	Enter the new password to be assigned, using a maximum of 12 characters. To leave the current password unchanged when you execute the utility, leave this field blank. To remove the current password and leave the files unprotected, enter two single quotes with no space between them ('''). The default is no change.
[Confirm each?]	Enter y to confirm the protection level assignment to each file. The default is no.

Establishing System and File Security

System security is a series of measures taken to prevent both damage to system software and unauthorized access to or alteration of protected data files.

You maintain security by granting appropriate system access to each user.

This section discusses in detail the tools BTOS provides to establish system and file security.

Tools for Establishing Security

The tools for establishing security include:

- BTOS file security system
- User signon files
- Command files

File Security

There are two elements of file security:

- Protection levels are assigned to files. A file's protection level determines which passwords a user must know, if any, to access a file.
- Passwords are assigned to devices, volumes, directories, and files.

Assigning Protection Levels

File protection levels control access to files. These levels determine the type of password, if any, that a user must enter to gain read-access or modify-access to a specific file.

You assign protection levels to files only. When you create a file, it automatically receives the default file protection level of the directory it is created in. This default level is set when the directory is created using the CREATE DIRECTORY command, or changed using the SET DIRECTORY PROTECTION command.

Eight protection levels are available. Table 5–1 lists the protection levels, their numbers, and the types of passwords required to read or modify files for each level.

Usually you use only three of these levels:

- Unprotected (level 15) allows all users unlimited file access and requires no password to read or modify the file.
- Modify Password (level 7) requires a password to modify the file, but does not require a password to read the file.
- Access Password (level 3) requires a password to read or modify the file.

Table 5–1 **BTOS Password Protection Levels**

		Password Required	
Protection Level	**Decimal Value**	**To Read**	**To Modify**
Unprotected	15	None	None
Modify password	7	None	Volume, directory, or file
Access password	3	Volume, directory, or file	Volume, directory, or file
Nondirectory password	51	Volume or file	Volume or file
Nondirectory modify password	23	None	Volume or file
Nondirectory access password	19	Volume, directory, or file	Volume or file
Modify protected	5	None	Volume or directory
Access protected	0	Volume or directory	Volume or directory
Read password	1	Volume, directory, or file	Volume or file

Passwords

Password protection is available at four levels:

- device
- volume
- directory
- file

Note: If you do not assign a password to a volume, the password protection of directories and files in that volume does not work.

The following paragraphs describe each password level.

Device Passwords. Device passwords are required for operations that work directly with a disk device, such as the MIVOLUME command or MBACKUP VOLUME command.

By default, device passwords are the same as the device name. For example, the device password for built-in disk d3 is d3. You can change device passwords by modifying the appropriate FP or DP configuration file or by customizing the FP or DP operating system. (Refer to the *XE500 BTOS Installation and Implementation Guide* and the *BTOS II Customizer Programming Guide.*)

Volume Passwords. A volume password controls access to all files on the volume. The volume password overrides directory or file passwords.

You can assign a volume password when initializing the volume with the MIVOLUME command. You can also assign a password or change an existing password by using the CHANGE VOLUME NAME command.

If you have assigned a volume password, users must enter it:

- in the MIVOLUME command form to initialize the volume
- in the CHANGE VOLUME command form to change the volume name or password
- in the CREATE DIRECTORY command form to create directories

In addition, by using the volume password as his or her signon password, a user can access all files in that volume. The user does not need to enter directory or file passwords to access those files.

Directory Passwords. You can use a directory password to restrict the access, creation, and renaming of files within a directory.

A directory password has no effect unless the volume containing the directory has a password.

You can assign a directory password when creating the directory with the CREATE DIRECTORY command. You can also assign a password or change an existing password by using the SET DIRECTORY PROTECTION command. You assign the password for the [sys]<sys> directory when you initialize the system disk through the mIVolume utility.

If you have assigned a directory password, users must enter it:

- in the SET DIRECTORY PROTECTION command form to set default protection levels for files in a directory
- in the REMOVE DIRECTORY command form to be able to remove a directory

If you have assigned a password to a directory, you can assign files in the directory a protection level that allows access with the directory password.

Note: It is recommended that you do not use the same password for a volume and a directory in that volume. This defeats the purpose of the multilevel passwords. To make a file or directory more accessible, you should change the file protection level.

File Passwords. A file password restricts access to a specific file, depending on the protection level you assign (refer to Assigning Protection Levels). A file password has no effect unless the volume and directory containing the file has a password.

You can assign a password to a file when you create the file using the CREATE FILE command (refer to the *BTOS II Standard Software Operations Guide*). You can also assign a password to an existing file using the MSET FILE PROTECTION command.

Accessing Password–Protected Files

A valid password can have a maximum of 12 characters, including alphanumeric characters, periods, and/or hyphens. Passwords should be easy to remember and unique to your system.

Users can access a password–protected file in two ways:

- entering an appropriate password as part of the file specification
- if their default password matches an appropriate password for that file

Passwords in a File Specification

You include a password in a file specification by following the file name with a caret (^) and the password. For example, the file specification [Personnel]<Engineering>Salaries^ABC includes the password ABC.

When a user enters a password as part of a file specification in a command form, each character of the password appears on the screen as a pound sign (#) to ensure confidentiality. Depending on the file protection level assigned to the file, a user must enter the volume, directory, or file password associated with that file to access the file. If a user does not supply an acceptable password, the system displays the error message Access denied.

User's Default Password

A user's default password matches the signon password when he or she initially signs on to the system. This password is defined in the user's signon file. A user can change his or her default password by using the LOGIN or PATH command (refer to the *BTOS II Standard Software Operations Guide*).

Depending on the file protection level assigned, the user's default password is compared with the file, directory, or volume password if no password is provided as part of a file specification.

User Signon Files

Each user must have a signon file to log into the system. The contents of this signon file determine, among other things:

- which BTOS commands the user can access
- which path (volume and directory) a user will be placed on after signing on
- which application a user enters after signing on
- the user's signon (default) password

An explanation of the signon file and how to create one for a user is provided in your *BTOS II Standard Software Operations Guide*.

Controlling Access to Master Utilities

You should limit user access to certain master utilities to prevent inadvertent misuse. You can control access in the following ways:

- creating customized user command files

  When you create a user's signon file, you designate a command file for the user. This command file lists all the commands that the user can execute. You can create customized command files for your users and thereby limit their access to selected commands. Refer to the *XE500 BTOS Installation and Implementation Guide* for information about creating command files.

- password–protecting the file [sys]<cmd>WsAdminAgent.txt

  If this file is password protected, only users whose default password matches a password appropriate for this file can execute the master commands listed in this file.

Guidelines for Establishing System Security

Note: The guidelines discussed in this subsection are intended to minimize accidental damage to one user's files by another user, and to make it difficult for a hostile user to read or to modify files in an unauthorized manner. There are no guarantees, expressed or implied, that these measures prevent a determined, hostile user from damaging system software or from accessing protected files.

When you first install your system software, no volume passwords or directory passwords are provided. To secure your system, you should implement passwords as soon as possible.

To simplify keeping track of the passwords, it is recommended you make all volume passwords a single password and all directory passwords a single password.

If you set your default password to a volume password, you can read and modify any file on that volume. Anyone else who knows the volume password can do the same. You should make certain that as few people as possible know the volume password.

System User Classes and Security Levels

Different types of users require different system resources. A data–entry clerk, for example, requires access to different resources than a word processing supervisor.

For the purposes of providing guidelines for establishing system security, this text assumes that you can divide users into three categories:

- single–program users
- command–subset users
- system administrator

The following subsections provide specific examples of how to implement security for each type of user.

Single–Program User

The single–program user uses one application, such as accounts receivable, data entry, or word processing. This user does not use system utilities. Thus, the only contact the user should have with the system is with the application being used and related data files. The user signon files are identical for single–program users who are using the same application.

As an example, suppose you want a group of users who are entering customer orders to run the order entry application program. You also want to restrict their access to the order entry database. Such a user's signon file might look like the following:

```
:SignonVolume:accounts
:SignonDirectory:orders
:SignonFilePrefix:
:SignonPassword:orderpass
:SignonExitFile:[sys]<sys>Signon.run
:SignonTextFile:
:SignonChainFile:orderentry.run
:ExecCmdFile:
```

This sets the user's path to volume accounts, in the orders directory, with a null file prefix.

Upon completion of a successful signon, the [accounts]<orders>orderentry.run application is run at the user's workstation. When the application terminates, the [Sys]<sys>Signon.run program executes, causing the signon form to display. Therefore, the user does not have access to the Executive command line and cannot execute any BTOS commands. Note also that the **:ExecCmdFile:** field specifies no command file.

To restrict user access to only the application program's run file and data base files, those files would be password-protected with the password orderpass.

Command Subset Users

Command subset users are users who must use system commands to perform their work, but who should not have access to commands that might accidentally damage system software or other user's files.

For example, suppose you have users who are doing data entry work and who must have access to all directories and files on a certain XE520 volume. In addition, they need to perform basic operations on those files, such as copying, deleting, renaming, backing up, and restoring. Such a user's signon file might look like:

```
:SignonVolume:specs
:SignonDirectory:sys
:SignonFilePrefix:
:SignonPassword:specspass
:SignonExitFile:[sys]<sys>Signon.run
:SignonTextFile:[sys]<sys>messageofday.txt
:SignonChainFile:[sys]<sys>Exec.run
:ExecCmdFile:[sys]<sys>specs.cmds
```

Upon signon, the user's path is the volume he or she is working in. The message of the day displays, followed by the Executive command line.

The user has access to only those commands included in the command file [sys]<sys>specs.cmds.

If the password for the volume specs is specspass, a user who uses this password when logging into the system has access to all files on the volume.

System Administrator

As system administrator, knowing all of the volume passwords in your system allows you to access all files in the system. Also, specifying the command file [sys]<sys>sys.cmds, which should contain all workstation and XE520 commands required to run your system, in your signon file ensures that you have access to all commands.

Protection Levels for System Files

Table 5–2 outlines suggested protection levels for system files, to ensure the security of the system.

Note: For the password–protection mechanism to be effective, only the system administrator should know the volume and directory passwords of the disk where any system files reside (normally, the XEBTOS system disk). Table 5–2 assumes this is the case.

Table 5–2 Suggested Protection Levels for System Files

File Type	Level	Explanation
User signon file	0	File cannot be modified or read. Ensures that users cannot read other users' signon passwords.
System run file	5	File can be read but not modified. Ensures that run files cannot be corrupted.
System data file	3	File can be read or modified only if the file password is known. Allows access to system data files for selected users.
System configuration file	5	File can be read but not modified. Ensures that system configuration files cannot be corrupted.
Operating system file	3	File can be read or modified only if the file password is known. Allows access to operating system files for selected users, such as a system programmer who is customizing an operating system.

Table 5–2 Suggested Protection Levels for System Files (continued)

File Type	Level	Explanation
Command file	0	File cannot be modified or read. Ensures that a user cannot read a command file and use the information to recreate that command using the NEW COMMAND command.
AdminAgent text file	0	File cannot be modified or read. Ensures that a user cannot modify the [sys]<cmd>WsAdminAgent.txt file, which controls access to all master commands.
Application programs	3	File can be read or modified only if the file password is known. Allows access to the file for selected users, such as an application programmer who has to maintain the application.

Managing File System Archives

A system hardware failure can make access to data stored on a disk impossible. To minimize the impact of such a failure, you should archive (back up) the system files periodically onto tape. Then, if a system failure prevents access to files, you can restore the files from your archive library.

You can back up an entire volume to archive files with the MBACKUP VOLUME or the MTAPE BACKUP VOLUME master command. You can also selectively back up files or directories with the MSELECTIVE BACKUP or the MTAPE SELECTIVE BACKUP command.

The MRESTORE and MTAPE RESTORE commands restore complete volumes, selected directories, or selected files from archive files.

Types of Archive Media

Depending on your system configuration, you can use one of the following types of tape to store your archive files:

- quarter–inch cartridge (QIC) tape
- half–inch tape

You can also store archive files on an XE520 built–in disk or a storage module device (SMD) disk, but this is strongly discouraged. If data on a disk that contains archive files becomes inaccessible due to hardware failure, you may not be able to recover the archive files.

It is strongly recommended that use tapes for archiving and that you keep two copies of archive files, in case one set of tapes becomes corrupted.

Using Archive Commands

When creating archive files, you should use the archive commands and not the copy commands.

The archive utilities provide a more reliable method of transcribing files to archive files. The archive utilities perform checking procedures to ensure the system has copied files properly; the copy utilities do not.

The MBACKUP VOLUME and MTAPE BACKUP VOLUME commands also verify the integrity of volume control structures of the volume you are backing up. If the system finds that the control structures are corrupted, the utility prompts you with a suggestion that you reinitialize the volume.

Archiving with Tapes

The following subsections describe how to back up XE520 volumes and files, and how to restore backed up files to a volume, using tapes.

The Backup Operation

The backup utilities do not perform one–to–one file copies from an XE520 volume file to tape. Instead, they copy groups of volume–based files into one archive file on the tape.

If you are backing up an entire directory, directory information (such as maximum number of files, number of sectors, default file protection level, and so on) is included in the archive file. This allows you to recreate the directory if you are restoring the backed–up files to a volume that does not have that directory.

Retensioning QIC Tapes

You should retension, or wind and rewind, a QIC tape under all of the following conditions:

- the first time you use it
- after every 8 hours of normal use
- after every 2 hours of extensive use in start/stop mode
- after a long period of disuse
- after exposure to temperatures above 113°F (45°C) or below 41°F (5°C)

To retension the tape, use the following procedure:

1 Insert the tape cartridge into the tape drive.

2 Enter **MQIC RETENSION** at the Executive command line and press GO.

The system retensions the tape.

Multiple Backups on One Tape

You can place multiple backups on one tape by filling in [tapexy]0, [tapexy]1, [tapexy]2, for half-inch tape or [qic]0, [qic]1, [qic]2, for QIC tape each time you use the MTAPE BACKUP VOLUME command. If you perform the backups in succession, you can use [tapexy] or [qic] to indicate that the backup files are written starting at the current file mark position of the tape.

Backup to Multiple Tapes

A large backup can require more than one tape. In that case, the last record on the tape indicates continuation to another tape. (The continuation must take place at the start of the second tape.)

If a second tape is needed, the backup utility prompts you to mount a new archive tape. The utility continues backing up volume files into that archive file.

The backup utility keeps track of the archive files by numbering them sequentially (each tape has a header record that contains the sequence number).

Reusing Archive Tapes

When backing up to multiple tapes, the backup utility checks to see if an archive file of the same sequence number exists on the tape. If one does, the utility prompts you for confirmation to overwrite the existing (old) archive file. If an archive file of the same sequence number does not exist, the utility attempts to create the archive file.

Backing up an XE520 Volume to Tapes

To back up an entire volume, you use the MTAPE BACKUP VOLUME command. The MTAPE BACKUP VOLUME command form gives you the following options:

- You can back up only the files created or modified after a specified date.

  This option lets you perform daily backup operations more quickly. For example, at the end of a work week, you can perform a backup of the entire volume. At the end of all other work days, you should back up only those files that have been modified after the date of the last full backup.

  If you must restore the volume, you restore the archive files of the weekly backup first. Then you restore the files of the daily backup, overwriting any duplicate files with the daily backup version.

- You can suppress the backup operation.

  This option allows you to verify volume control structures without actually performing a backup.

- You can suppress the verification of volume control structures.

  This option allows you to back up files without verifying volume control structures; this shortens the time needed to complete the operation.

  You should suppress the verification pass only when you perform a full backup, and only if you are going to reinitialize and restore the volume immediately after the backup.

- You can delete an existing archive file.

  Using this option causes the utility to overwrite an existing archive file with the new archive file without prompting for confirmation.

- You can specify a print file you want the backup report copied to.

  It is recommended that you use this option to obtain a printed copy of the files you successfully backed up.

- You can display volume control structure information.

  You can use this information to analyze defects in the volume control structures.

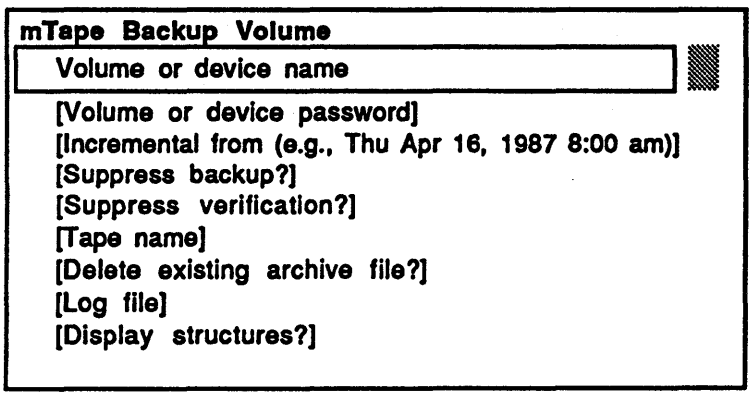
To execute the MTAPE BACKUP VOLUME command, you enter **MTAPE BACKUP VOLUME** at the Executive command line and press RETURN. The system displays the command form shown in figure 6–1. Table 6–1 lists the fields in the MTAPE BACKUP VOLUME command form.

The command form requires the name of the volume (or the corresponding disk name) you want to back up and the name of the tape file you want to use as the archive file. (Refer to your *XE500 BTOS Operations Guide* for information about tape name conventions.) All other fields are optional. When you have filled in the appropriate fields of the command form, you press GO.

The mTape Backup Volume utility begins by prompting you to mount the specified archive tape. After you confirm the archive tape is mounted, the utility opens the archive file on the tape. If an archive file of the same name already exists, the utility prompts you to choose whether to overwrite the existing archive file.

Once the archive file is opened, the utility begins backing up the files from the volume. If more than one tape is required, the system prompts you to mount the next tape in the sequence.

Figure 6–1 **MTAPE BACKUP VOLUME Command Form**



When all files have been backed up, the utility reports the number of files successfully and unsuccessfully backed up. If a memory sector on the volume is corrupt, the utility inserts zeros into the corrupted portion of the backed up file.

Unless you have overridden the volume verification option in the command form, the utility then performs a verification of the volume structures and reports the volume structure status.

Note: To verify the files were backed up properly, it is strongly recommended that you use the **[List files only?]** option of the MTAPE RESTORE command. Refer to the subsection on the MTAPE RESTORE command.

Table 6–1 **MTAPE BACKUP VOLUME Command Form Fields**

Field	Action/Explanation
Volume or device name	Enter the name of the volume to be backed up. You can use the device name of the disk drive the volume is stored on rather than the volume name.
[Volume or device password]	Enter the password (if one exists) for the volume you are backing up. If you specify a device name in the first field, the device password (if one exists) should be used instead.
[Incremental from (e.g., Thu Apr 16, 1987 8:00 pm)]	Enter the date you want files backed up from. Only files modified on or after the specified date and time are backed up. If you do not specify a time, the time is assumed to be the beginning of the specified day. If you leave this field blank, all files in the volume are backed up.
[Suppress backup?]	Enter yes to verify the integrity of volume control structures without actually performing a backup. The results of the verification display on the screen. To perform a backup, leave this field blank.

Table 6–1 **MTAPE BACKUP VOLUME Command Form Fields** (continued)

Field	Action/Explanation
[Suppress verification?]	Enter yes to back up a volume without verifying the integrity of its control structures. You should suppress the verification pass only when you are performing a full backup and only if you are reinitializing the volume with the mIVolume utility immediately following the backup. Otherwise, the integrity of the volume cannot be assured. In any case, errors encountered during backup display on the screen. To have volume control structure verification performed, leave this field blank.
[Tape name]	This field is not optional. You must enter the name of the tape file you are backing up to. (Refer to your *XE500 BTOS Operations Guide* for information about tape name conventions.)
[Delete existing archive file?]	Enter yes to automatically overwrite an existing archive file on the archive tape. If you leave this field blank, mTape Backup Volume prompts you to choose whether you want to overwrite an existing archive file.
[Log file]	To create a print file of the mTape Backup Volume report, enter a file name. You can then view this file at your screen or print it. You should not place the file on the volume you are backing up. To send the listing to a printer automatically and not save the print file, enter a print queue name, enclosed in square brackets. By default, the listing displays only on the screen.
[Display structures?]	Enter yes to have a detailed analysis of the volume control structures included in the backup report. System programmers can use the information from this analysis for analyzing defects in the file system. If you do not want a detailed analysis of the volume control structures included in the backup report, leave this field blank.

Backing up Selected Files to Tapes

To back up selected files to tape, you use the MTAPE SELECTIVE BACKUP command. The MTAPE SELECTIVE BACKUP command form contains a **File List** field where you can specify one or more files in a volume. The default path is your current working volume and directory. You can use wild card characters in the file specification to specify groups of files.

The MTAPE SELECTIVE BACKUP command form gives you the following options:

- You can back up only files from the file list created or modified after a specified date.
- You can be prompted to confirm the backup of each file from the file list.
- You can delete an existing archive file on the tape.

  Using this option causes the utility to overwrite an existing archive file with the new archive file without prompting for confirmation. The existing archive file must have the same name (that is, the same sequence number) as the new archive file.
- You can specify a printable log file to which the backup report is copied.

  It is recommended that you use this option to obtain a printed copy of the files successfully backed up during the backup operation.

To execute the MTAPE SELECTIVE BACKUP command, you enter **MTAPE SELECTIVE BACKUP** at the Executive command line and press RETURN. The system displays the command form shown in figure 6–2. Table 6–2 lists the fields in the MTAPE SELECTIVE BACKUP command form.

You enter the tape file name you want to use as the archive file in the **[Tape name]** field. (Refer to your *XE500 BTOS Operations Guide* for information about tape name conventions.) All other fields are optional. After you fill in the appropriate fields of the command form, you press GO.

The mTape Selective Backup utility begins by prompting you to mount the specified archive tape. After you confirm the archive tape is mounted, the utility opens the archive file on the tape. If an archive file of the same name already exists, the utility prompts you to choose whether you want to overwrite the existing archive file.

Figure 6–2 **MTAPE SELECTIVE BACKUP Command Form**

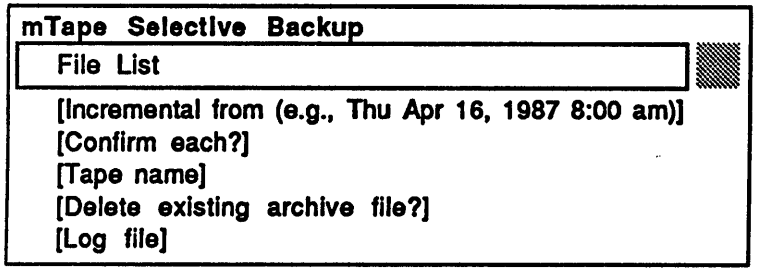


Table 6–2 **MTAPE SELECTIVE BACKUP Command Form Fields**

Field	Action/Explanation
File List	Enter a file, a list of files, or a directory to be backed up. The entry can list single files or designate sets of files (using wild card characters in the file specification).
[Incremental from (e.g., Thu Apr 16, 1987 8:00 pm)]	Enter the date from which files are to be backed up. Only files modified on or after the specified date and time are backed up. If you do not specify a time, the time is assumed to be the beginning of the specified day. If you leave this field blank, all specified files are backed up.
[Confirm each?]	Enter yes to be prompted to confirm the backup of each file. To back up all files specified, leave this field blank.
[Tape name]	This field is not optional. You must enter the name of the tape file you are backing up to. (Refer to your *XE500 BTOS Operations Guide* for information about tape name conventions.)
[Delete existing archive file?]	Enter yes to automatically overwrite an existing archive file on the archive tape. If you leave this field blank, mTape Selective Backup prompts you during the backup to choose whether you want to overwrite an already existing archive file.

Table 6–2 **MTAPE SELECTIVE BACKUP Command Form Fields** (continued)

Field	Action/Explanation
[Log file]	To create a print file of the mTape Selective Backup report, enter a file name. You can then view this file on your screen or print it. To automatically send the listing to a printer and not save the print file, enter a print queue name, enclosed in square brackets. By default, the listing displays only on the screen.

After opening the archive file, the utility begins backing up the specified files. If more then one tape is required, the system prompts you to mount the next tape in the sequence.

When you have backed up all files, the utility reports the number of files successfully backed up. If a memory sector on the volume is corrupt, the utility inserts zeros into the corrupted portion of the file.

Note: To verify that the files were backed up properly, it is strongly recommended that you use the **[List files only?]** option of the MTAPE RESTORE command. Refer to the subsection on the MTAPE RESTORE command.

Restoring Files from Tapes

To restore files to an XE520 volume from archive tapes, you use the MTAPE RESTORE command.

The MTAPE RESTORE command form allows you to exercise the following options:

- You can specify the files to be restored.
- You can specify the destination of the files being restored and can restore a file using a different file name than the original file's name.
- You can automatically overwrite a file at the destination volume and directory with the same file name as the file being restored.

- You can be prompted to confirm the restore operation of each file.
- You can specify an archive file sequence number other than 1 with which to start the restore operation.

Caution: You must consider two points about this option:

First, directory information is maintained only on the first file of the archive files (that is, the archive file appended with .01). If a restore operation begins with an archive file other than this first file, mTape Restore creates any required directory, allots it 10 sectors of memory, and sets protection levels for all files as unprotected. If 10 sectors are insufficient, you must explicitly create this directory before beginning the restore operation.

Second, during the backup operation, a file is sometimes split across archive files. Such a file is restorable only if the two archive files it is split across are restored in sequence.

- You can merge the contents of the file being restored with an existing file of the same name.

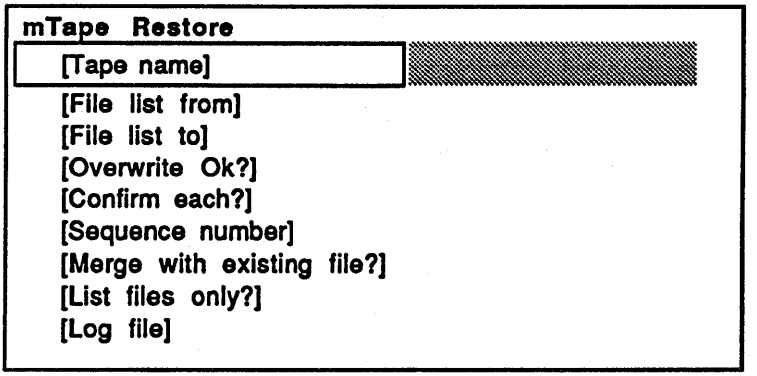
  This option is used when overwriting an existing destination file with its corresponding backed-up version. If this option is exercised and sectors of the backed-up file were filled with zeros when it was backed up, the corresponding sectors in the existing file are not overwritten.

  If this option is not used, the backed up file is restored as is.
- You can override the restore operation and list only the files contained on the archive media. You can use this option to check that files were backed up properly or to view the contents of the archive media without actually restoring files.
- You can specify a printable log file the backup report is copied to.

  It is recommended that you use this option to obtain a printed copy of the files successfully restored.

To execute the MTAPE RESTORE command, you enter **MTAPE RESTORE** at the Executive command line and press RETURN. The system displays the command form shown in figure 6–3. Table 6–3 describes the fields in the MTAPE RESTORE command form.

Figure 6–3 **MTAPE RESTORE Command Form**



You enter the tape file name of the archive file in the **[Tape name]** field. (Refer to your *XE500 BTOS Operations Guide* for information about tape name conventions.) All other fields are optional. After you fill in the appropriate fields of the command form, you press GO.

The mTape Restore utility begins by prompting you to mount the specified archive tape. After you confirm the archive tape is mounted, the utility begins restoring files according to the parameters specified in the command form. Directories are created as required. Also, each file is restored with the same characteristics (for example, protection level, password, and so on) that it had when it was backed up.

When you are restoring files from multiple tapes, the tapes must be inserted in the same sequence as the backup operation. The utility prompts when you must insert the next tape in the sequence.

Table 6–3 **MTAPE RESTORE Command Form Fields**

Field	Action/Explanation
[Tape name]	This field is not optional. You must enter the name of the tape file you are restoring from. (Refer to your *XE500 BTOS Operations Guide* for information about tape name conventions.)
[File list from]	Enter a list of the files you want to restore. The file list specification must be of the form <dirspec>filespec, where wild card characters can be used in the directory or file specification. If you leave this field blank, all the files in the archive file are restored. Note that you cannot leave this field blank without also leaving the [File list to] field blank.
[File list to]	Enter a list of the files you want the backed–up files restored to. The file list specification must be of the form [volname]<dirspec>filespec, where the volume name is optional and wild card characters can be used in the directory or file specification. If you do not include a volume name, files are restored to the current working volume. If you enter a file list in this field, you must properly match the wild card characters used in the file list spec in the [File list from] field. For example, if you enter <*>* in the [File list from] field, you can enter <*>* in the [File list to] field; the files would be restored to the same directories and with the same names as they were backed up. You could also enter <*>*.jcl in the [File list to] field to alter the names of the restored files, or <d*>* to alter the names of the restored directories. If you leave this field blank, all the files are restored to the directories in the current working volume with the same names as those the files were backed up from. In all cases, directories are created as necessary from the information stored in the archive files.

Table 6–3 **MTAPE RESTORE Command Form Fields** (continued)

Field	Action/Explanation
[Overwrite Ok?]	Enter yes to automatically overwrite any existing destination files of the same name as files being restored. The existing file is deleted before the backed up file is restored. Enter no if you do not want any existing destination files overwritten. If such a file is encountered, the system prompts you that it is not carrying out the restore operation for that file. Leave this field blank to have the system prompt you to choose whether you want an existing file overwritten.
[Confirm each?]	Enter yes to be prompted to confirm the restoring of each file. To restore all specified files, leave this field blank.
[Sequence number]	Enter the sequence number of the restore operation's beginning archive file. For example, if you are restoring from archive tapes and want to start the restore with the second archive tape, enter a 2. The default is 1, the first of the archive files.
[Merge with existing file?]	Enter yes to bypass overwriting the sectors of the destination file if the corresponding sectors of the file on the archive medium being restored were unreadable when backed up. If you leave the field blank, the backed up file is restored as is (that is, with zeros in any corrupted areas).
[List files only?]	Enter yes to get a list of the files stored on the archive media without actually restoring any. Leave this field blank if you want to restore files.

Table 6–3 **MTAPE RESTORE Command Form Fields** (continued)

Field	Action/Explanation
[Log file]	To create a print file of the mTape Restore report, enter a file name. You can then view this file on your screen or print it. To send the listing to a printer automatically and not save the print file, enter a print queue name, enclosed in square brackets. By default, the listing displays only on the screen.

How Tape Utilities Manage Backup and Restore Tape Errors

Each tape block is numbered sequentially on the tape. The sequence number is surrounded on one side by a check word, and on the other side by the tape file number. If an error occurs when the tape block is written, it is rewritten (further down the tape) with the same sequence number.

As the tape is read, each sequence number is monitored. Duplicate sequence numbers are ignored and not passed to the backup program as data.

mTape Backup Volume and mTape Selective Backup do not append data to a tape with unreadable tape or file headers. However, if unreadable tape or header records are encountered while mTape Restore or mTape Selective Backup is being executed, the tape is scanned until a valid data block is found (that is, if the data block conforms to the format of tape backup data block).

Scheduling Backup Operations

The backup interval you choose should reflect the amount of new data entered into the system data files. If you cannot afford the loss of a single day's data, you should back up daily. Some administrators find weekly backups sufficient. The recommended procedure is to back up both daily and weekly.

To back up a volume both daily and weekly, use the following procedure:

1 At the end of each working day, back up the volume using the incremental backup option.

  In the **[Incremental from]** field of the backup command form, enter the date following the day you performed the last weekly backup.

  You can use the same archive tape for each daily backup, overwriting the previous day's archive files. Or, for added security, you can use different archive tapes for each daily backup. This would allow you to recreate the state of the volume for any day that week.

2 On the last working day of the week, perform a full backup using a weekly archive tape.

This procedure insures that, in the event of unrecoverable volume corruption, you will only lose the current day's changes to that volume. You can restore the volume or move it to another disk by restoring from the weekly archive tape and then from the daily archive tape. When you restore from the daily tape, you use the overwrite option of the restore command so the latest versions of the files are restored.

Examples of Backing Up and Restoring Files

The following subsections include examples of backing up and restoring XE520 files. These examples are meant to show just a few of the ways you can use these commands.

Examples of Backing Up Files

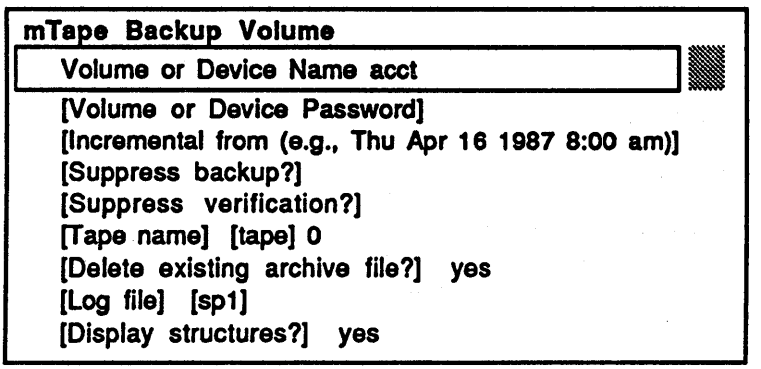
This subsection shows examples of using the MBACKUP VOLUME, MSELECTIVE BACKUP, MTAPE BACKUP VOLUME, and MTAPE SELECTIVE BACKUP commands.

Example 1

The command form in figure 6–4 shows a backup using the MTAPE BACKUP VOLUME command with the following parameters:

- The name of the volume to be backed up is acct. This volume has no password. The entire volume is being backed up.
- The archive file is being stored under the tape file name [tape]0, which is the first tape file mark of the half–inch tape in tape drive 0.
- Volume control structures are to be verified and a detailed listing of the results is to be reported.
- The existing archive file on the tape will be overwritten automatically.
- The backup report will be printed automatically at a printer servicing the spl print queue.

Figure 6–4 **Sample MTAPE BACKUP VOLUME Command Form**

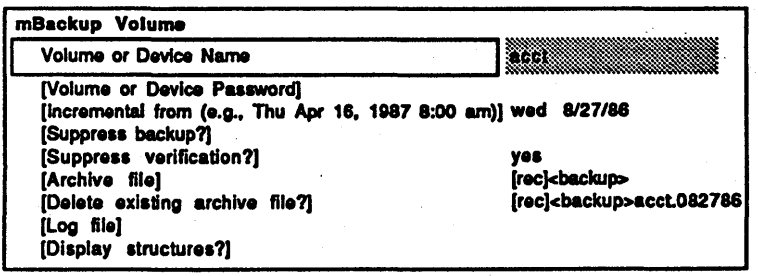


Example 2

The command form in figure 6–5 shows a backup operation using the MBACKUP VOLUME command with the following parameters:

- The name of the volume to be backed up is acct. This volume has no password.
- Only those files created or modified since the beginning of the day August 27, 1986, are to be backed up.
- Volume control structures will not be verified.
- The archive file root name [rec]<backup> will be used. That is, archive files will be [rec]<backup>.01, [rec]<backup>.02, and so on.
- The system will display a prompt to overwrite existing archive files in [rec]<backup> directory.
- The backup report will be copied into the file [rec]<backup>acctlog.082786.

Figure 6–5 **Sample MBACKUP VOLUME Command Form**

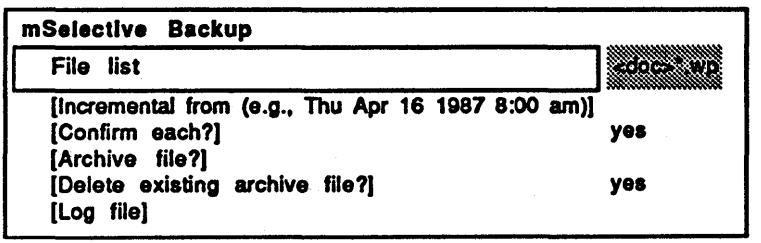


Example 3

The command form in figure 6–6 shows a backup using the MSELECTIVE BACKUP command with the following parameters:

- All files in the <doc> directory that end in .wp will be backed up.
- The default archive file names are to be used. Existing archive files will be overwritten automatically.
- The system will prompt the user to confirm the backup of each file.

Figure 6–6 **Sample MSELECTIVE BACKUP Command Form**

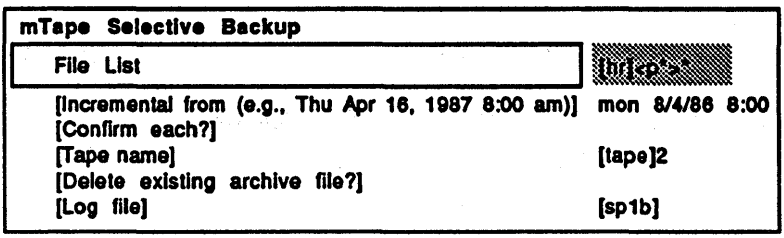


Example 4

The command form in figure 6–7 shows a backup using the MTAPE SELECTIVE BACKUP command with the following parameters:

- The archive file is stored under the tape file name [tape]2, which is the third tape file mark of the half–inch tape in tape drive 0.
- The file list includes all files in directories beginning with the letter p on the [hr] volume.
- Only those files created or modified since 8:00 am, August 4, 1986, are to be backed up.
- The backup report will be printed automatically at a printer servicing the splb print queue.

Figure 6–7 **Sample MTAPE SELECTIVE BACKUP Command Form**



Sample Restore Operations

The following subsections describe some sample restore operations.

Restoring All Files on the Archive Tape

If you want to restore all files on the archive tape to the current working volume and do not want to use any of the command options, you enter the appropriate restore command at the Executive command line and press GO.

To restore all files on the archive tape to an XE520 volume other than the current working volume, you fill in the file list fields as follows:

[File list from] <*>*

[File list to] [volname]<*>*

volname is the destination volume (or device) name.

Restoring Selected Files on the Archive Tape

You can use wild card characters in the file list specification for the **[File list from]** field of the restore commands to select files or groups of files to restore.

For example, the file list entries

[File list from] <tom>*

[File list to] <tom>*

restore all files backed up from the directory <tom> to the directory <tom> on the current working volume.

The file list entries

[File list from] <t*>*.wp

[File list to] <t*>*.wp

restore all files ending in .wp and backed up from directories beginning with t to the corresponding directories on the current working volume. Any of those directories that do not exist on the current working volume will be created with information stored in the archive files.

You can also have backed-up files restored to directories and files with different names. For example, the file list entries

[File list from] <*>*.wp

[File list to] <*t>*.wp-version1

restore, to directories with t appended to the original directory names, all files ending in .wp, and add the suffix -version1 to the file names.

Duplicating Release Software

No utility duplicates release software distributed on tape and preserves the original format. It is therefore strongly recommended that you store release tapes in a safe place.

Managing Secondary Partitions

An XE520 processor's memory can be divided into regions called partitions. A program the processor is running can be assigned a partition to execute in. The program restricts its instructions and data to this assigned area. The processor can then operate in a multitasking mode, with several programs running on the processor at the same time. The programs share processing time.

There are three types of partitions:

- a system partition, where the operating system resides

  No applications run in the system partition.
- a primary partition, where any services that require user interaction should run

  To ensure optimum system performance, some system services must be run in a primary partition. A processor's primary partition is set up automatically during system start-up. Each processor must have one, and only one, primary partition.
- a secondary partition, where applications, including some system services, that do not require user interaction can run

This section discusses how to manage secondary partitions on an XE520 processor. Partition management includes the following tasks:

- creating a secondary partition
- installing a system service into a secondary partition
- removing a secondary partition
- obtaining status information about partitions

Creating Partitions during System Start–up

You can have secondary partitions automatically created and loaded with applications during system start–up. You do this by adding appropriate run statements to the target processor's initialization file. This subsection describes the procedure.

Note: When you are creating secondary partitions, the remaining memory in the primary partition must be at least as large as the new secondary partition and must also have room for the CLI to be loaded after Create Partition has executed. This means that if you add a secondary partition to a processor, the processor must have 768 kB of memory.

Some of the standard release system services are preconfigured to be installed in secondary partitions. If new system services are to be installed after your system is delivered, you can create secondary partitions for them to enhance overall system efficiency.

To have a secondary partition for a processor created automatically during system start–up, you use the Editor to include the following run statement in the processor's initialization file:

$run [sys]<sys>CreatePartition.run,size,name

size is an optional parameter that determines the size of the partition when it is created. The parameter value is the size of the partition, in kilobytes (kB), followed by the letter k. The size of a partition is limited only by available memory. If this parameter is not specified, the default value 200k is used.

In addition to the space required by an application program, the partition needs 1.5 kB to store system data structures associated with it. mInstall Server requires that the minimum size of a secondary partition be 6 kB.

name is an optional parameter that determines the name of the secondary partition. If this parameter is not specified, the default name of the partition is backgroundnn, where nn is 00 the first time the utility is invoked on the processor, 01 the second time, and so on.

It is strongly recommended that partition names contain only alphanumeric characters. Only names with alphanumeric characters are displayed by other programs. The partition name must not exceed 12 characters.

The partition will be created the next time the system is booted.

You can obtain status information about any secondary partition by executing the MPARTITION STATUS command. Refer to Getting Partition Status Information.

Installing Applications

Note: If you are installing an application into a secondary partition on a processor, that processor must be running the AdminAgent. Refer to the *XE500 BTOS Installation and Implementation Guide* for information about how to configure the AdminAgent.

Installing System Services from the Executive

To install an application into a secondary partition from the BTOS Executive, you use the MINSTALL SERVER command.

To execute the MINSTALL SERVER command, use the following procedure:

1 Enter **MINSTALL SERVER** at the Executive command line and press RETURN.

  The system displays the MINSTALL SERVER command form shown in figure 7–1. Table 7–1 lists the fields in the MINSTALL SERVER command form.

2 In the **Partition** field, enter the name of the partition the application is to run in.

3 In the **Server run file** field, enter the name of the application's run file.

  The command form includes additional optional parameter fields you can use to pass parameters to the application.

4 When you have filled in the appropriate fields of the command form, press GO.

  The system prompts you for the processor on which the partition resides.

5 Enter the appropriate processor name (for example, FP00 or CP01) and press RETURN.

You have installed the application into the secondary partition as specified.

Figure 7–1 **MINSTALL SERVER Command Form**

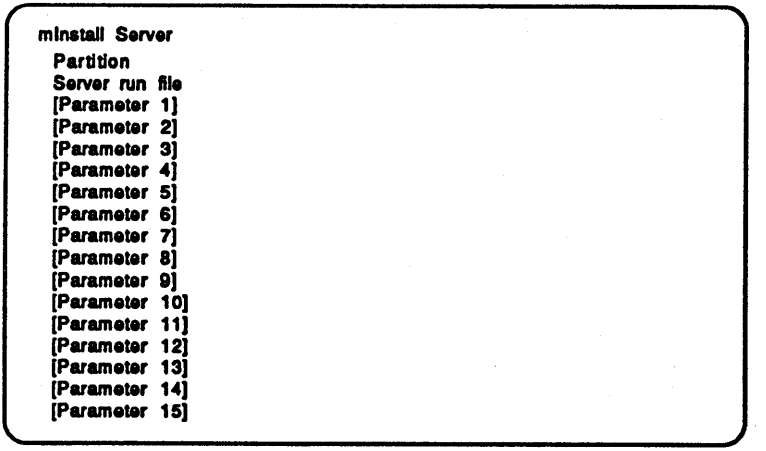


Table 7–1 **MINSTALL SERVER Command Form Fields**

Field	Action/Explanation
Partition	Enter the name of the partition the application is to run in.
Server run file	Enter the name of the application's run file.
[Parameter n]	Enter application–related parameters that you want to pass to the application in these fields.

Installing System Services during System Start–Up

You install system services during system start–up by making appropriate additions to the processor's .jcl file. If the system service uses ConvertToSys, you add a run statement specifying the system service. If the system service does not use ConvertToSys, you add the run statements for Create Partition (see Creating Partitions during System Start–Up) and mInstall Server.

Note: If you write your own system services, you should have them install with ConvertToSys rather than with Create Partition and mInstall Server.

The mInstall Server run statement has the form

$run [sys]<admin>mInstallServer.run,PartName,SvrRunFile

PartName is the name of the secondary partition and **SvrRunFile** is the run file, including any parameters, of the service to be installed. The application will be loaded into the partition the next time the system is booted.

If the server fails to install during system start–up, an error message is logged in the system log, [sys]<sys>log.sys. You can examine the message with the PLOG command.

You can remove these applications by deleting the appropriate run statements from the processor initialization files.

Synchronizing the Installation of System Services

When you install system services on more than one board and need to ensure that one system service is installed before another, you use the Sync program. Sync matches up pairs of installation operations, suspending operations on a board until installation operations on the matched board have begun.

If, for example, you are installing the Queue Manager on FP00 and the Spooler on CP00 and CP01, the JCL files are as follows:

File Processor FP00:

```
$Run [Sys]<Sys>InstallQMgr.run,y,30
$Run [Sys]<Sys>Sync.run,1
$Run [Sys]<Sys>Sync.run,2
```

Cluster Processor CP00:

```
$Run [Sys]<Sys>Sync.run,1
$Run [Sys]<Sys>MSpoolerMgr.run
```

Cluster Processor CP01:

```
$Run [Sys]<Sys>Sync.run,2
$Run [Sys]<Sys>MSpoolerMgr.run
```

Processing begins on all three processors at once: FP00 installs Queue Manager; CP00 runs Sync 1, and waits; CP01 runs Sync 2, and waits.

After installing Queue Manager, FP00 runs Sync 1. This allows CP00, the processor that has run the matching Sync 1, to continue processing. CP00 installs the Spooler.

After running Sync 1, FP00 runs Sync 2. This allows CP01, the processor that has run the matching Sync 2, to continue processing. CP01 installs the Spooler.

Monitoring Synchronized System Service Installation

You use the SYNC STATUS command to monitor synchronized installation of system services. You can also use it to free a processor that is waiting for a matched Sync that cannot execute.

To monitor installation or free a processor with the SYNC STATUS command, use the following procedure:

1. At the Executive command line, enter **SYNC STATUS.**
2. In the **[Request numbers]** field, enter the numbers you have used for sync requests (1 for Sync 1, 2 for Sync 2, and so on).

3 Press GO.

The system displays the status of each of the sync numbers. When a processor is waiting for another to run a matching Sync, the system displays Wait. When the processor receives notification that the matching Sync has run, the system displays Free.

4 If an installation fails on one processor and another processor is waiting, move the cursor to the line on the screen that shows the waiting process and press F10 (Free).

The sync program is overridden and the waiting processor no longer waits.

Removing Secondary Partitions

You cannot remove secondary partitions through the Executive.

To prevent a secondary partition from being created and loaded automatically during system start–up, you delete the appropriate run statement from a processor initialization file. Refer to Creating Partitions during System Start–Up.

Getting Partition Status Information

If you want status information about a secondary partition, you use the MPARTITION STATUS command.

Status information includes:

- logical processor name and slot number
- partition name and state (vacant, locked, batch, and so on)
- partition size in kB (value is in decimal)
- available memory in partition in kB (value is in decimal)
- current run file for partition
- exit run file for partition
- indication of whether the exit run file has a password
- priority of the current run file

If you select the verbose option of the command, the following additional information displays:

- partition handle
- partition low–bound segment address
- partition long–lived memory low segment address
- partition long–lived memory high segment address
- partition short–lived memory low segment address
- partition short–lived memory high segment address
- partition high–bound segment address

To execute the MPARTITION STATUS command, you enter MPARTITION STATUS at the Executive command line and press RETURN. The system displays the command form shown in figure 7–2. Table 7–2 lists the fields in the MPARTITION STATUS command form.

In the **CPU name** field, you enter the specification for the processor the partition is on (for example, FP00, CP01, and so on). You enter **all** if you want to list the status of all secondary partitions in the system.

The command form has three optional fields that allow you to:

- create a print file of the listing or have the listing printed automatically
- suppress page pause messages so that the listing does not stop with each screen of text
- use the verbose option to list the partition handle and memory segment information

To create a print file of the status listing, you enter a file name in the **[Print file]** field. You can then view this file at your workstation or print it. To automatically send the listing to a printer and not save the print file, you enter a print queue name, enclosed in square brackets, in the **[Print file]** field. By default, the listing only displays at the workstation.

To suppress the Executive page pause feature that occurs with each screen of text, you enter **no** in the **[Pause between pages?]** field. By default, the page pause feature is used as the status listing is displayed.

To have the partition handle and memory segment information included in the status listing, enter **yes** in the **[Verbose?]** field. By default, this information is not included in the status listing.

When you have filled in the appropriate fields of the command form, press GO. The status listing displays at the workstation.

MPARTITION STATUS Command Form

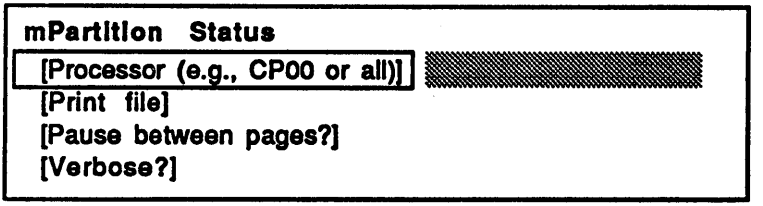


Table 7–2 **MPARTITION STATUS Command Form Fields**

Field	Action/Explanation
[Processor (e.g. CP00 or all)]	Enter the specification for the processor the partition is on (for example, FP00, CP01, and so on). Enter all if you want to list the status of all secondary partitions in the system. The default is the current processor.
[Print file]	To create a print file of the status listing, enter a file name. If the file already exists, the report is appended to the existing file. You can view the file on your screen or print it. To automatically send the listing to a printer and not save the print file, enter a print queue name, enclosed in square brackets field. The default is Vid; the listing displays on the screen.

Table 7–2 **MPARTITION STATUS Command Form Fields** (continued)

Field	Action/Explanation
[Pause between pages?]	If you enter yes, mPartition Status pauses every time it displays or echoes 23 lines. When mPartition Status pauses, it displays Press NEXT PAGE/RETURN to continue. To continue, press NEXT PAGE if you ran the mPartition Status utility through the mCli utility. If you ran the mPartition Status utility any other way, press RETURN. Enter no to suppress the Executive page pause feature that occurs with each screen of text. You cannot abort the program you are running by pressing CANCEL or FINISH. By default, the page pause feature is used as the status listing is displayed. The default is no pause.
[Verbose?]	Enter yes to have the partition handle and memory segment information included in the status listing. The default is no. If you leave this parameter blank, mPartition Status lists only the name and slot number of the processor executing the mPartition Status utility and the name and slot number of each processor specified.

Managing Cluster Activity

This section describes the MDISABLE CLUSTER and MRESUME CLUSTER commands, which allow you to manage cluster activity.

- The MDISABLE CLUSTER command allows you to temporarily suspend cluster activities (for example, when you are installing software on the XE520).
- The MRESUME CLUSTER command allows you to resume cluster activities after a temporary suspension.

Disabling Cluster Lines

Occasionally, you must suspend cluster activities temporarily. For example, many applications require you to disable the cluster when you install software.

When you execute the MDISABLE CLUSTER command, XEBTOS:

- sends status code 46 and the message Master workstation going down to all workstations
- stops communicating with the cluster workstations
- closes all open files

If a user enters a request for an file or system service while cluster activities are suspended, XEBTOS returns status code 6 and the message Master workstation not running.

Note: Any applications running at the workstations must take appropriate action when they receive status code 46, Master workstation going down.

To activate the MDISABLE CLUSTER command, you enter MDISABLE CLUSTER at the Executive command line and press RETURN.

The system displays the MDISABLE CLUSTER command form shown in figure 8–1. The command form fields are described in table 8–1.

Figure 8–1 **MDISABLE CLUSTER Command Form**

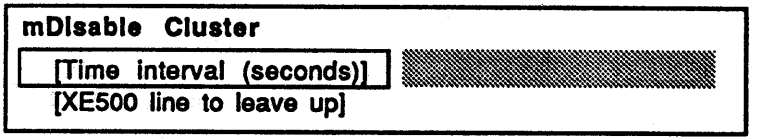


Table 8–1 **MDISABLE CLUSTER Command Form Fields**

Field	Action/Explanation
[Time interval (seconds)]	Specify the number of seconds that you want the system to delay before disabling the cluster lines. The default is 0. A delay lets workstation users finish activity before their cluster lines are disabled.
[XE500 line to leave up]	Enter the cluster line number your workstation is attached to. All other lines except the one specified are disabled when the command is executed. Refer to table 8–2 for a list of the line numbers.

If you want the system to wait before disabling the cluster, you specify the length of the delay, in seconds, in the **[Time interval (seconds)]** field.

To allow your workstation to continue communicating with the XE520, you must specify one cluster line to remain active by entering the line number in the **XE500 line to leave up** field. Line numbers are assigned to Cluster Processor (CP) RS–422 Cluster Communications port connectors by order of CPs in the system. Table 8–2 lists line number assignments for Cluster Communications ports in an XE520.

Table 8–2 **MDisable Cluster Line Numbers**

Line Number	Cluster Communications Port
1	CP00, Cluster 1
2	CP00, Cluster 2
3	CP01, Cluster 1
4	CP01, Cluster 2
5	CP02, Cluster 1
6	CP02, Cluster 2
7	CP03, Cluster 1
8	CP03, Cluster 2

Caution: Make sure of the number of the line your workstation is connected to. If you specify the wrong line number, your workstation is disabled. If you specify a line number not applicable to your system configuration, you disable all workstations.

If all workstation cluster lines become disabled, you can enable them only by running the mResume Cluster utility from a CLI port (refer to section 9) or by rebooting the system.

Resuming Cluster Activity

When you are ready to restore cluster activity, you execute the MRESUME CLUSTER command by entering **MRESUME CLUSTER** at the Executive command line and pressing GO. XEBTOS resumes communications with the cluster workstations.

The MRESUME CLUSTER command has no command form.

Note: Files that were open on a cluster workstation when the cluster was disabled are not automatically reopened when the line is reenabled. For the application running at the workstation to resume without user intervention, the application must reopen the files.

Using the Command Line Interpreter

The XEBTOS Command Line Interpreter (CLI) interprets and executes CLI commands to XEBTOS. The CLI acts like the BTOS Executive for the following specific administrative and implementation tasks:

- When a processor is booted at system start-up, BTOS uses the CLI to interpret the run statements in the processor's initialization file.
- The XEBTOS Debugger is run through the CLI. The Debugger is used to test application programs running on XE520 processors and to analyze processor memory dumps. Refer to the *XE500 BTOS Debugger Operations Guide* for more information about the Debugger.
- The CLI can install and run applications on processors during normal system operations.

CLI commands are ASCII text lines. These text lines are submitted to the CLI through processor initialization files, from an RS-232-C serial terminal connected to a CP or TP CLI port, or from a workstation running the mCdtIO or mCli utility. The rules for CLI statements are similar to those for the BTOS Job Control Language (JCL) used by the BTOS Batch system. If you are familiar with Batch JCL, you should note the following differences between the two sets of rules.

- When CLI commands are entered interactively, the dollar sign ($) at the beginning of Batch JCL statements is unnecessary.
- The CLI has no notion of jobs or job control. Therefore, job statements are ignored. You use the PATH command to specify the path.
- The CLI always executes in the primary application partition. SysIn and SysOut currently can communicate only with the terminal.
- The CLI ignores log statements.

How CLI Works

The CLI runs in the primary partition of each BTOS processor. Only one program can execute in a BTOS partition at a time, so when CLI runs another program, it sets the BTOS exit run file to indicate its own run file, [sys]<sys>Cli.run. The successive instances of the CLI can be considered as a single program; they maintain a context file to maintain continuity.

To show how CLI works, it is helpful to trace what happens after a processor's operating system is loaded during system bootup.

1 Once the processor's operating system is loaded into memory, BTOS executes its initial program, the CLI, in the primary application partition.

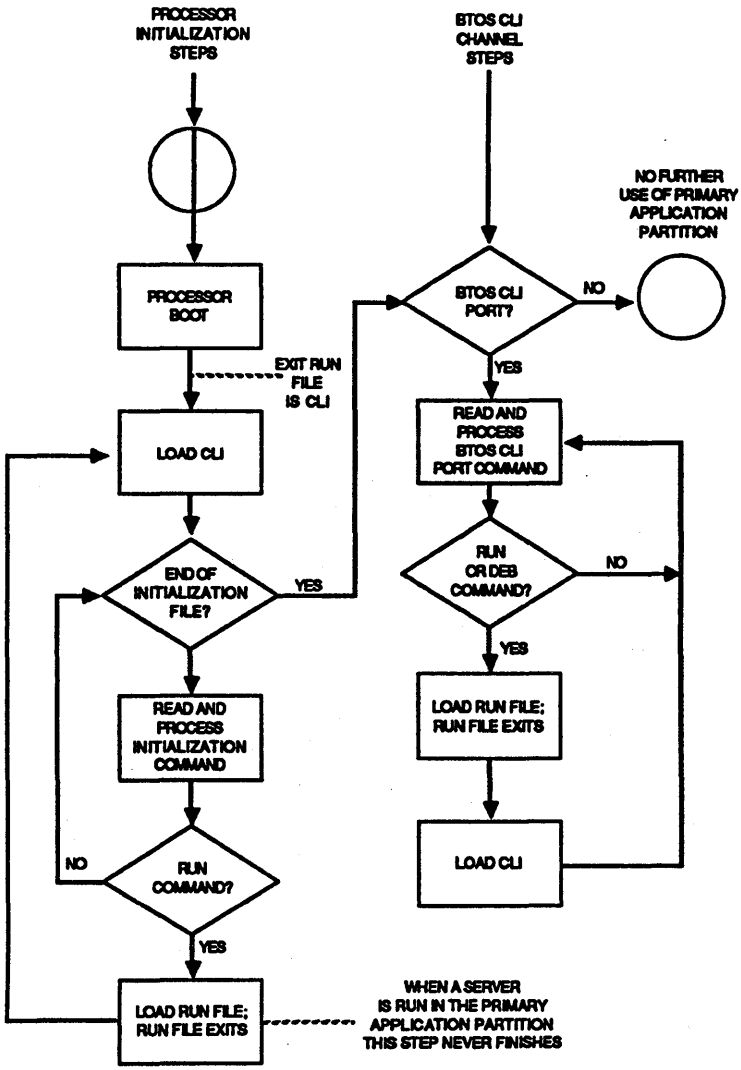
2 The CLI executes the run statements in the processor's initialization file. If a run statement is to execute another program, the execution includes three steps:
   - The CLI sets the exit run file to be its own run file.
   - The CLI has BTOS run the other program, overlaying the CLI in the primary application partition.
   - When the other program terminates, BTOS executes the exit run file, reloading the CLI into the primary application partition.

3 If the processor is configured with a CLI port, the CLI executes any commands entered from that channel. If a command is to execute another program, the execution follows the same procedure described in step 2.

Figure 9–1 is a flow chart representing this process.

If a program is to run concurrently with the CLI, the CLI commands must install the program in a secondary application partition. Refer to section 7 for more information about running applications in secondary partitions.

Figure 9–1 **Processor Initialization File Execution**



Communicating with the CLI

There are three ways to communicate with the CLI:

- with command statements in a processor initialization file
- interactively using an RS-232-C serial terminal connected to a CP or TP CLI port

  You define a CP or TP CLI port by adding the phrase **connect=ctos** to the end of the RS-232-C channel's entry in the processor configuration file. There can only be one CLI port per CP or TP. For a CP, it must be channel 3; for a TP, it must be channel 10.
- interactively from a workstation using the mCdtIO or mCli utility

  You invoke the mCdtIO or mCli utility from the BTOS II Executive by using the MCDTIO or MCLI command.

Note: The destination processor you want to run a CLI session with must be running Cli.run in its primary partition, unless you are using the MCLI command. (If you are using mCli, the AdminAgent must be running in the destination processor's primary partition.) You can check whether Cli.run is running in the processor's primary partition by running mPartition Status on that processor (refer to section 7).

Cli.run will be running in the processor's primary partition unless the application installed in the primary partition never exits. Some BTOS servers that are executed in the primary partition through processor initialization files do exit: for example, the Queue Manager. Others, such as AdminAgent, the printer spooler manager, and the Tape Server, do not exit.

Using Processor Initialization Files

Each processor normally has an initialization file. Processor initialization file names take the form [sys]<sys>initXpnn.jcl, where Xpnn is the processor designation (for example, FP00, CP02).

The initialization file is a text file that has a line entry for each system service to be run on the processor. Each line entry in an initialization file is a run statement that loads and activates the system service's run file on the processor.

For more information about processor initialization files and the services you can run on them, refer to the *XE500 BTOS Installation and Implementation Guide.*

Using CLI Ports

Each CP or TP can have one CLI port. To designate an RS–232–C channel as the CLI port, you add the phrase **connect=ctos** to that channel's entry in the processor's configuration file.

If you run CLI from this channel, an RS–232–C serial terminal must be connected to the channel.

The terminal hardware must be set with the following line parameters:

- 9600 baud
- 8 data bits
- one stop bit
- no parity

When the terminal is turned on, CLI prompts for input with a dollar sign ($).

Commands executed through the CLI port can only be run on the processor to which the CLI port is connected. However, by executing the run statement for the mCdtIO or mCli utilities, you can have programs execute on other processors. This allows you to execute commands on processors other than CPs and TPs. The next subsections discuss the mCdtIO and mCli utilities.

Using the mCdtIO Utility

The mCdtIO utility provides the same level of interaction with any processor that a CLI port provides for a CP or TP. For example, using the mCdtIO utility, you can connect a debugger to an FP or execute a set of CLI initialization commands on an SP, with full interactive capabilities.

You can invoke the mCdtIO utility in two ways: from the Executive, through the MCDTIO command; or from the CLI, through a CLI run statement that specifies the mCdtIO run file. The MCDTIO command form has two optional fields that allow you to specify:

- whether a log file of the MCdtIO session is to be created or automatically printed at the end of the session
- whether you are to be prompted to continue with each screen of text

  By default, the utility does not use page pauses.

Executing MCdtIO Through the Executive

To execute the mCdtIO utility through the BTOS Executive, use the following procedure:

1. Enter **MCDTIO** at the Executive command line and press RETURN.

   The system displays the MCDTIO command form shown in figure 9–2. Table 9–1 lists the fields in the MCDTIO command form.
2. In the **Processor** field, enter the name of the target processor (for example, FP00, CP02).
3. To exercise either of the available options, creating a log file or requesting page pauses, press RETURN and fill in the appropriate field.
4. When you have filled in the appropriate fields of the command form, press GO.

   The utility begins running and the CLI dollar sign ($) prompt displays.

Figure 9–2 **MCDTIO Command Form**

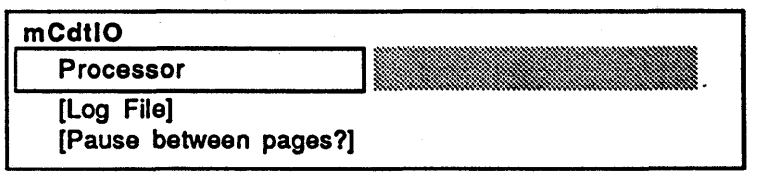


Table 9–1 **MCDTIO Command Form Fields**

Field	Action/Explanation
Processor	Enter the name of the target processor (for example, FP00, CP02).
[Log file]	Enter the name of the file the log of the MCDTIO session is to be written to. If the log file already exists, the existing file is deleted and the new file is created in its place. If the log file does not exist, it is created. If you leave this field blank, the interaction is not recorded. To have the log automatically printed out and not save the log file, enter the name of a printer queue, enclosed in square brackets. If you do not name a print file, the information is only sent to the screen. The default is video.
[Pause between pages?]	Leave the field blank if you do not want a system prompt at the end of each screen of text. If you enter yes, the McdtIO utility pauses every time it displays or echoes 23 lines. When the McdtIO utility pauses, it displays Press <NEXT PAGE>/<RETURN> to continue. Press NEXT PAGE if you ran the McdtIO utility through the Mcli utility. Press RETURN if you ran the McdtIO utility any other way. The default is no.

Note: Next page prompts appear during the MCdtIO utility if you specify yes in the **Suppress pause between pages?** field of the SCREEN SETUP or VIDEO command or yes in the **Pause between pages** field of the MCDTIO command.

Therefore, if you request page pauses both ways, you will be prompted twice for Next Page during the mCdtIO utility.

To eliminate page pauses, you specify no page pauses in both the SCREEN SETUP or VIDEO command and the MCDTIO command.

mCdtIO pagination does not permit you to abort the program you are running by pressing CANCEL or FINISH.

Executing mCdtIO Through a CLI Port

To execute the mCdtIO utility from a serial terminal connected to a CLI port, you enter the run statement for the mCdtIO utility at the CLI prompt as follows:

$run [sys]<sys>mCdtIO.run,cpu,log,pause

where the parameters cpu, log, and pause are as described for the MCDTIO command.

Terminating an mCdtIO Utility Session

To terminate an mCdtIO utility session from a workstation, you press CODE–D twice.

To terminate an mCdtIO utility session from a serial terminal connected to a CLI port, you press CONTROL–Z once or CONTROL–D twice.

Using the mCli Utility

The mCli utility provides the same level of interaction with any processor that a CLI port provides for a CP or TP. For example, with the mCli utility, you can execute a set of CLI initialization commands on an SP, with full interactive capabilities.

Note: The Debugger cannot be executed during a CLI session initiated through the mCli utility.

To invoke the mCli utility, you enter **MCLI** at the Executive command line and press GO. The utility begins running on the processor that the AdminAgent is running on, and the CLI dollar sign ($) prompt displays.

Terminating an mCli Utility Session

To terminate an mCli utility session from a workstation, you press CODE–D twice.

CLI Command Syntax

The CLI command syntax comprises a basic command form and conventions for continuation lines, comments, and call parameters.

Command Form

A CLI command has the form

$command params

$ is the prompt displayed by interactive CLI; for JCL files, it indicates that the line in the file is a CLI command.

command is the name of the CLI command. You can enter the command in upper or lower case. If the command is not one of the standard CLI commands, it is assumed to be an implicit call to a JCL file (refer to Calling a JCL File for Execution). You must separate the parameters from the command with a space, not a comma.

params is a parameter list. You must separate parameters from each other with commas; spaces directly before or after parameter commas are ignored.

A parameter is identified solely by its position in the parameter list. If you are not specifying certain optional parameters, you must still include a comma in the command for each parameter left out, unless you specify no subsequent parameters. For example, if an application has, in order, a required parameter and two optional parameters, and you want to specify the required parameter and only the second optional parameter, you use the run statement

$run [sys]<sys>application.run,param1,,param3

To specify only the required parameter, you use the run statement

$run [sys]<sys>application.run,param1

Parameters are passed literally, including the case of letters.

Special Characters

The CLI attaches special meaning to dollar sign ($), ampersand (&), percent sign (%), semicolon (;), comma (,), parentheses [()], and backslash (\). To include a special character literally in a command, you precede it with a backslash, as in the following example:

```
$run Loan.run, \$150\,000, 12\%
```

To keep spaces and commas as part of a parameter, you enclose the parameter in single quotes. For example,

```
$run Newname.run, 'J. C. Ryke, Jr.'
```

To run a program with subparameters, you enclose the group of parameters in parentheses [()]. The parameters within the parentheses are passed as a single parameter with subparameters. For example, in the following run statement for the Delete utility, p2, p3, and p4 are files being specified for the **File list** parameter of the utility:

```
$run [sys]<sys>Delete.run,(p2 p3 p4),y
```

Continuation Lines

In a text file, if you must continue a command on another line, you end the preceding line of the command with an ampersand (&). When CLI is used interactively, the CLI prompts for the continuation line with an ampersand instead of a dollar sign.

For example, the following two run statements are functionally identical, but the second one uses a continuation line:

```
$run [sys]<sys>Example.run,p1,p2,p3,p4

$run [sys]<sys>Example.run,p1,p2,&
p3,p4
```

Comments

To insert a comment that the CLI is to ignore, you insert a semicolon (;) before the comment. The CLI ignores anything after a semicolon on a line. For example:

```
$return ; sort completed

; initcp01.jcl
$run mpartitionstatus.run
```

Call Parameters

A JCL file executed by the CLI CALL command or an implicit call can access the additional parameters specified following the JCL file name. The additional parameters in the JCL file are numbered from zero and are preceded by a percent sign (%0, %1, %2, and so on).

An example is the following JCL file called [sys]<sys>pcopy.jcl:

```
; pcopy - copy a file from one prefix to another
; format: pcopy file,prefix1,prefix2
$run [sys]<sys>copy.run,%1%0,%2%0
```

The following three CLI commands are equivalent (the third command line is an implicit call):

```
$run [sys]<sys>copy.run,[d2]<sys>dbms,[d3]<sys>dbms
$call [sys]<sys>pcopy.jcl,dbms,[d2]<sys>,[d3]<sys>
$pcopy dbms,[d2]<sys>,[d3]<sys>
```

File Name Conventions

File names in CLI commands follow the standard BTOS file name conventions. Refer to section 3 or to the *BTOS II Standard Software Operations Guide* for an overview of BTOS file name conventions.

By default, the CLI path is set to [sys]<sys>. You can use the PATH command to change the current path during a CLI session. (Refer to Changing the Path.)

Using CLI Commands

The command name, the first word of a CLI command, specifies the action to be executed. Table 9–2 lists the CLI commands. Some commands listed in the table are ignored by the XE520 CLI; they serve no function but are listed for compatibility with Batch JCL.

In addition to these commands, you can create your own commands that will implicitly call JCL files; refer to Calling JCL Files for Execution.

Table 9–2 **CLI Commands**

Command	Use
Cancel	Terminates execution of a called JCL file.
CancelOnError	Indicates to the CLI to terminate processing of a JCL file if an error is returned.
ContinueOnError	Indicates to the CLI to continue processing a JCL file even if an error is returned.
Deb	Loads and debugs an object code run file.
End	Signals the end of a JCL file.
Job	Ignored.
Log	Ignored.
Path	Sets the BTOS working path.
Prefix	Sets the prefix of a JCL file being called through an implicit call.
Return	Terminates execution of the JCL file currently being processed.
Run	Runs an object code run file.
Suffix	Sets the suffix of a JCL file being called through an implicit call.

Executing a Run File

To load and execute a run file at a processor, you use the RUN command. A run file is a loadable object file produced by BTOS Link. By convention, such files have names ending in .run.

The RUN command must be followed by the name of the run file to be executed. The run file can, in turn, be followed by the parameters that apply to it. For example, the following command causes the Files utility to be executed on the directory [d2]<acct>:

```
$run [sys]<sys>Files.run, [d2]<acct>*, y, [spl]
```

If the program examines its command name, the first parameter should consist of two subparameters: the run file name and the command name, in that order. For example,

$run ([sys]<sys>myprogram.run, 'param1'),param2

When the run file execution is complete, BTOS reloads the CLI and execution proceeds with the next CLI command.

Calling JCL Files for Execution

There are two ways to call a JCL file for execution by the CLI:

- You can directly call a JCL file, using the CALL command. The CALL command must be followed by a space and the name of the JCL file. The JCL file name can then be followed by a comma and any parameters to be passed to the JCL file during execution.
- You can call a JCL file implicitly by entering the root name of the JCL file name. By default, the JCL file name must have the form [sys]<sys>rootname.jcl, where rootname is the name to be used as the CLI command. The root name can then be followed by a space and any parameters to be passed to the JCL file during execution.

  The default prefix, [sys]<sys>, and the default suffix, .jcl, can be changed using the PREFIX and SUFFIX commands.

If you are passing parameters to the JCL file from a call command, their target must be defined in the JCL file using call parameters.

A called JCL file can include call commands; these are nested calls. Calls can nest to any depth, provided there is room left in the context file.

An example of a call command is

$call [sys]<sys>Bill.jcl,x0,x1,x2

An example of an implicit call that is functionally identical to the previous call command is

$Bill x0,x1,x2

The contents of the JCL file [sys]<sys>Bill.jcl could be

$run [sys]<sys>billing.run,%0,%1,%2

To change the default prefix or suffix of an implicit call command, you use the PREFIX or SUFFIX command. This command must be executed before the implicit call statement. The prefix or suffix defined by a PREFIX or SUFFIX command is effective until the next instance of the PREFIX or SUFFIX command.

To use one of these commands, you enter the command name followed by a space and the new prefix or suffix to be used. The following example allows the JCL file called [d2]<acct>Bill.oldjcl to be called implicitly using the root name Bill:

$prefix [d2]<acct>

$suffix .oldjcl

$Bill

Ending a JCL File

You enter the END command as the last line of a JCL file to indicate where execution should end. The END command takes no parameters. It can be omitted without any effect, except if the file is also to be executed by the Batch system.

Terminating Execution of JCL Files

To terminate the processing of the currently executing JCL file, you use the RETURN command. This command has no parameters and terminates only one level of nesting.

To terminate the processing of the currently executing JCL file and all nested calls, you use the CANCEL command. After terminating all calls, the CLI is reloaded. If CLI had been reading a processor initialization file, it then begins execution of that file again. If interactive mode was being used, CLI then prompts for another command.

By default, the execution of a JCL file continues with the next command if an error is returned while the file is being processed. You can have CLI terminate the execution of a JCL file when an error is received by entering the CANCELONERROR command in the file. The CancelOnError state remains effective until a CONTINUEONERROR command is executed.

For example, the following JCL file causes execution to terminate only if an error is returned while the Copy utility is running.

$run [sys]<sys>Rename.run,[d2]<acct>*,[d2]<sales>*

$cancelonerror

$run [sys]<sys>Copy.run,[d2]<sales>*,[d3]<customers>*

$continueonerror

$run [sys]<sys>RemoveDirectory.run,[d2]<acct>

The RETURN, CANCEL, CANCELONERROR, and CONTINUEONERROR commands do not take parameters.

Changing the Path

To change your current working path during a CLI session, you use the PATH command. The PATH command has the format

$path node,vol,dir,file,pass

node is specified only if the system is running BNET, **vol** is the volume or disk device name, **dir** is the directory name, **file** is the file name, and **pass** is a password, if required. You should not use square brackets ([ ]) with the volume or disk device name, angle brackets (<>) with the directory name, or a caret (^) with the password.

The following example specifies a path of [d3]<bt> with the password security (no node or file is specified):

$path ,d3,bt,,security

The default path is {local}[sys]<sys> with no default password.

The new path is effective until changed by another PATH command.

Loading a Run File during a Debugger Session

To load a run file while the Debugger is activated, you enter the **DEB** command. The .cnf file for each processor you want to run Debugger on must contain the string

Debugger

at boot time.

Refer to your Debugger documentation. Execution, if any, is under Debugger control.

CLI Status Messages

The following is a list of CLI status messages, with explanations as needed.

Bad command name or no such command file exists.

Command file name:filename

The command file name given in a call statement does not exist. Note that any unrecognized CLI command is assumed to be an implicit call.

Bad yes/no parameter.

The specified yes/no parameter is invalid. Specify y or n.

Cannot open Context file in $ or current directory.

The Context file is used to save the state information parameters when JCL command files nest.

Cannot open specified command file.

Error in formatting date/time, error = x.

Error when accessing JCL file, error = x.

Fatal error encountered.

Termination status code:x

The currently running program had a fatal error. Refer to the *BTOS II System Status Codes Reference Manual* for the meaning of the status code.

Syntax error on current command line.

The file specification is invalid.

Unbalanced quotation marks in the **Parameters** field.

Monitoring XE520 Processor Activity

Two master commands, MHISTOGRAM and MPSTAT, generate listings you can use to monitor processor activity. This section describes how to use these master commands.

Monitoring a Processor's CPU Activity

To optimize an application, it is helpful to know where in the application most of the processing activity is occurring. The system programmer can then concentrate on optimizing that area of the application.

The MHISTOGRAM command generates a listing that indicates the level of activity for specified areas of a processor's central processing unit (CPU) memory. You can then isolate the memory address ranges where an application is spending the most CPU processing time.

The mHistogrammer utility displays a series of prompts through which you determine the target processor, address range, and number of samples to histogram.

To monitor an application's CPU time, use the following procedure:

1 Use the MPARTITION STATUS command with the verbose option to find out the address range the application is executing in. (Refer to section 7 for more information about the MPARTITION STATUS command.)

  Note the target address range, using paragraph numbers. (Refer to the *BTOS II Language Development Reference Manual* for more information on memory paragraphs.)

2 Install the mHistogrammer monitoring program on the target processor. The run file for the mHistogrammer monitoring program is [sys]<sys>mHistoProc.run.

  You can install the monitoring program in a primary or secondary partition. Refer to section 7 for more information on how to install an application into a processor.

3 Execute the MHISTOGRAM command by entering **MHISTOGRAM** on the Executive command line.

To exercise either of the available options, creating a log file or suppressing page pauses, press RETURN and fill in the command form. Figure 10–1 shows the MHISTOGRAM command form. Table 10–1 describes the MHISTOGRAM command form fields.

Press GO.

The mHistogrammer utility prompts for the processor to histogram.

4 Enter the processor designation (for example, FP00, CP01) of the processor the application is running on. Press RETURN.

The utility prompts for the starting paragraph number, in hexadecimal (hex).

5 Enter the starting paragraph number, in hex, and press RETURN. You must use four digits (for example, 0040 for paragraph 40).

The utility prompts for the ending paragraph number, in hex.

6 Enter the ending paragraph number, in hex, and press RETURN. You must use four digits (for example, 0040 for paragraph 40).

The mHistogrammer utility prompts for the bucket size, by paragraphs, in hex.

Buckets are the areas the targeted address range is divided into. Each bucket in the range is monitored for CPU activity. The mHistogrammer generates a listing of the buckets, by address range, and the percent of CPU time spent in each bucket.

7 Enter the size, in hex, of a bucket, and press RETURN.

The resulting number of buckets [(EndPara – StartPara + 1)/BucketSize] must be an integer from 1 to 512. If the bucket size is too small, the number of buckets exceeds 512, and the utility picks a size for you.

The utility prompts for the number of clock ticks to histogram.

8 Enter the number of CPU clock ticks for the sample interval and press RETURN.

This number represents the number of times you want the mHistogrammer utility to interrupt processing and check to see the address of the instruction that the CPU is currently executing.

For FPs, DPs, and SPs, the interrupt rate is 100 Hz. Therefore, if you want to sample for one second, enter **100**; for 10 seconds, **1000**; and so on.

For CPs and TPs, the interrupt rate is 1302 Hz. Therefore, if you want to sample for approximately one second, enter **1300**; for approximately 10 seconds, **13000**; and so on.

Figure 10–1 **MHISTOGRAM Command Form**

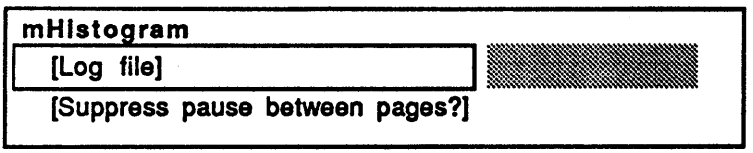


Table 10–1 **MHISTOGRAM Command Form Fields**

Field	Action/Explanation
[Log file]	Enter the name of the file the mHistogrammer report is to be written to. If the log file already exists, the existing file is deleted and the new file created in its place. If it does not exist, the log file is created. To have the report automatically printed and not saved, enter the name of a printer queue, enclosed in square brackets. If you do not name a print file, the information only displays on screen.

Table 10–1 **MHISTOGRAM Command Form Fields** (continued)

Field	Action/Explanation
[Suppress pauses between pages?]	The default is no; leave this field blank to have the system prompt you before each screen of data when the MHistogrammer report displays. Enter yes to suppress the Next page prompts. The report will scroll forward to its end when it displays.

After you enter the number of clock ticks and press RETURN, the histogramming utility executes and a generates a report. The report lists the following information:

- the number of slots in the array (that is, the number of buckets)
- the total number of ticks counted
- the number of ticks for which the CPU executed an instruction in the specified address range, and this number expressed as a percentage of the total number of ticks counted
- the range of each bucket, by paragraph
- the number of ticks for each bucket (that is, the number of times the CPU was found to be executing an instruction in that bucket's address range)
- the number of ticks in each bucket expressed as a percentage of (1) the number of ticks in the specified range and (2) the number of total ticks counted

After displaying the report, the mHistogrammer utility prompts you to choose whether to do another histogram. You enter **yes** and press RETURN to run through the prompts again and specify new parameters. You enter **no** and press RETURN to return to the Executive.

Figure 10–2 shows a sample report for two sets of parameters.

Figure 10–2 **Sample mHistogrammer Report**

```
XE500 Histogrammer, Version x.x

Processor to histogram (e.g. Tp00, Fp1,  Cp, etc.):  cp00
Starting Paragraph <hex>:  0000
Ending Paragraph   <hex>:  8000
Bucket Size (paragraphs)  <hex>:  100
Clock ticks to histogram :  3000

There are 97 slots in the array.

Total Ticks     :        3000
Ticks in Range:          3000  (  100.0 %)

SA                       TICKS          PERCENT
                                  IN RANGE     TOTAL
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
0000  -  00FF              10          0.3 %        0.3 %
0100  -  01FF            2287         76.2 %       76.2 %
0700  -  07FF              12          0.4 %        0.4 %
0800  -  08FF              30          1.0 %        1.0 %
0900  -  09FF             179          5.9 %        5.9 %
0A00  -  0AFF             122          4.0 %        4.0 %
0B00  -  0BFF             127          4.2 %        4.2 %
0C00  -  0CFF             233          7.7 %        7.7 %

Repeat histogramming ? y

Processor to histogram (e.g. Tp00, Fp1,  Cp, etc.):  cp000
Starting Paragraph <hex>:  0A00
Ending Paragraph   <hex>:  3000
Bucket Size (paragraphs)  <hex>:  100
Clock ticks to histogram :  5000

There are 39 slots in the array.

Total Ticks     :        5000
Ticks in Range:           781  (   15.6 %)

SA                       TICKS          PERCENT
                                  IN RANGE     TOTAL
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
0A00  -  0AFF             195         24.9 %        3.9 %
0B00  -  0BFF             186         23.8 %        3.7 %
0C00  -  0CFF             400         24.9 %        8.0 %

Repeat histogramming ? y
```

Monitoring Processor Intercommunications

XE520 processors communicate through a facility called Inter–CPU Communication (ICC). You can use the MPSTAT command to generate ICC statistics for all FPs, DPs, SPs, CPs, and TPs in the system.

Statistics generated by the MPSTAT command are listed by processor and are compiled for a specified time frame. They include:

- the number of Y and Z blocks allocated and currently available
- the number of requests made to the processor
- the number of requests made by the processor
- the amount of data, in kilobytes, copied into the processor while handling input requests
- the amount of data, in kilobytes, copied out of the processor while handling output requests
- the number of Y blocks used

Optionally, you can request disk I/O statistics instead of ICC statistics for FPs and DPs. Disk statistics are listed by processor and physical disk unit number. This information includes:

- the number of I/O blocks (IOBs) allocated and currently available
- the number of local and remote reads and writes to the disk
- the number of seeks performed on the disk
- the number of requests made by the processor

The MPSTAT command form has three optional fields that allow you to specify:

- Whether you want disk I/O statistics generated instead of ICC statistics
- the time interval to compile the statistics for
- the number of times to repeat the compilation and reporting of statistics

Executing the MPSTAT Command

To execute the MPSTAT command to generate only ICC statistics for all processors for one 10-second interval, enter MPSTAT in the Executive command field and press GO. The mPStat utility generates a table of statistics.

To use any of the optional fields, use the following procedure:

1 Enter **MPSTAT** at the Executive command line and press RETURN.

The system displays the MPSTAT command form shown in figure 10–3. Table 10–2 lists the fields in the MPSTAT command form.

2 In the **[ICC or Disk]** field, enter **disk** if you want to generate disk I/O statistics instead of ICC statistics.

3 In the **[Intervals]** field, enter the time interval, in seconds, for which you want statistics compiled.

4 In the **[Repetitions]** field, enter the number of times to repeat the compilation and reporting of statistics.

5 When you have filled in the appropriate fields of the command form, press GO.

The system generates the statistics as specified.

Figure 10–3 **MPSTAT Command Form**

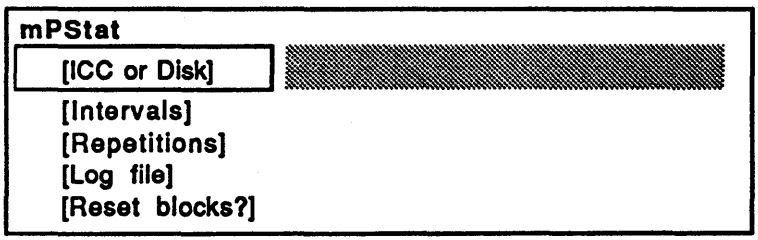


Table 10–2 **MPSTAT Command Form Fields**

Field	Action/Explanation
[ICC or Disk]	Enter disk if you want to generate disk I/O statistics (these statistics apply only to FPs and DPs). The default is ICC; to generate ICC statistics for all processors in the system, leave this field blank.
[Intervals]	Enter the time interval, in seconds, for which you want statistics compiled. The maximum value is 65000.
[Repetitions]	Enter the number of times to repeat the compilation and reporting of statistics. The maximum value is 65000.
[Log file]	Enter the name of the file the report is to be written to. To have the report automatically printed and not saved, enter the name of a printer queue, enclosed in square brackets. If you do not name a print file, the information only displays on screen.
[Reset blocks?]	Enter yes to reset the minimums for X, Y, and Z blocks (for the ICC option) or I/O blocks (for the Disk option) to the minimum for the current interval. The default value, no, preserves the values for the period since the system last rebooted.

Figure 10–4 shows two sample listings: one of a system's ICC statistics and one of disk statistics for the same system. Each row represents the statistics for one processor. Table 10–3 explains the meaning of each column in the ICC listing. Table 10–4 explains the meaning of each column in the disk I/O listing.

Note that in the ICC statistics, you can calculate the average request size by dividing Kb Copied In by Requests In.

Figure 10–4 **Sample mPStat Statistics for ICC and Disk**

```
                    Zblk                Yblk                Requests        Kb Copied
                                                                            Yblks
Board   Cur  Min  Max   Cur  Min  Max    In   Out    In    Out   Used
FP00    29   17   30    22   20   23     446  39     41.1  35.5  439
CP00    83   80   90    6    6    6      1    9      0.0   0.1   0
DP00    27   26   28    4    4    4      6    0      0.1   0.1   0

                    Local               Remote                      IOB
Board   Unit        Reads  Writes       Reads  Writes  Seeks  Cur  Min  Max
FP00    1           27     8            353    0       42     25   13   25
DP00    2           29     36           0      54      22     25   23   25
```

Table 10–3 **mPStat ICC Statistics**

Column	Meaning
Board	processor name
Cur	number of blocks currently available
Min	lowest number of blocks ever available
Max	number of blocks initially allocated
Requests In	number of requests the processor received
Requests Out	number of requests the processor made
Kb Copied In	total amount of data, in kilobytes, copied into the processor while handling input requests
Kb Copied Out	total amount of data, in kilobytes, copied out of the processor while handling output requests
Yblks Used	number of Y blocks the processor used

Table 10–4 **mPStat Disk I/O Statistics**

Column	Meaning
Board	processor name
Unit	physical unit number of the disk drive Statistics are printed only for devices that have performed some I/O during the sample period.
Local Reads	number of disk sectors (one sector is 512 bytes) read into local memory
Local Writes	number of disk sectors written from local memory
Remote Reads	number of sectors written into remote memory, usually data or virtual memory
Remote Writes	number of sectors written from remote memory
Seeks	total number of seeks done on the disk
Cur IOB	number of I/O blocks currently available
Min IOB	lowest number of I/O blocks ever available
Max IOB	number of I/O blocks initially allocated

Using the MPSTAT Command Statistics

You can diagnose two types of problems using the MPSTAT command statistics:

- Insufficient resource allocation can be indicated by the presence of zeros in the Yblk and Zblk Min columns. To correct this, you increase the number of Y or Z blocks for that processor.

  Insufficient resource allocation can also be indicated if the number of Yblks used is a large fraction of the number of Requests In. To correct this, you can increase the size of a Z block. If you change the size of a block on any processor, you must make the change on all processors. If the Z block size increases, more requests can use Z blocks instead of Y blocks.

- Improper resource balance is indicated if kilobyte rates differ greatly for equivalent processors. For example, if the system has two CPs, workstations should be balanced between them until kilobyte rates are approximately equal over time.

To modify the size or number of Y or Z blocks, you modify the Y or Z block allocation entries in the processor's configuration file or customize that processor's operating system. For information about modifying Y and Z blocks entries in processor configuration files, refer to the *XE500 BTOS Installation and Implementation Guide*. For information about customizing processor operating systems, refer to the *BTOS II Customizer Programming Guide*.

Running the System in Different Operating Modes

You can boot up XEBTOS using different sets of system configuration files and processor BTOS versions. The set of files used depends on the position the keyswitch is turned to and defines the operating modes the system can run in.

There can be four sets of system configuration files:

- a customized set used when the keyswitch is turned to MANUAL

  The system configuration file names in this set include .m (for example, [sys]<sys>Master.m.cnf).
- the set used when the keyswitch is turned to REMOTE

  The system configuration files names in this set include .r (for example, [sys]<sys>Master.r.cnf). This set is provided in the standard software release. It is used to bring the system up in restricted mode.
- a customized set used when the keyswitch is turned to NORMAL

  The system configuration files names in this set include .n (for example, [sys]<sys>Master.n.cnf).
- the default set, which is the standard set of system configuration files created through the MBTOS Config utility

  The file names in this set do not have an identifying letter (for example, [sys]<sys>Master.cnf). These files are used if the corresponding .r, .m, or .n files do not exist.

The system files included in these sets are:

- the master configuration file
- processor configuration files
- processor initialization files
- a CP, DP, and SP version of BTOS

  For master FPs and DPs, only one version of the master operating system, [sys]<sys>SysImage.sys, is used, regardless of the keyswitch position.

What Happens At Boot Time

By turning the keyswitch at the XE520 base enclosure from STOP to MANUAL, REMOTE, or NORMAL, you initiate the XE520 boot-up process. The master FP first runs a series of self-diagnostic tests and loads the master OS, [sys]<sys>SysImage.sys. It then attempts to read the set of system files associated with the position to which the keyswitch is turned. These files include the identifying characters .m for MANUAL, .r for REMOTE, and .n for NORMAL. For each type of system file, if the system does not find the corresponding keyswitch-related file, it uses the default version of the file.

As part of the software installation procedure, the set of system configuration files whose file names include .r is installed. This set is used to bring the system up in restricted mode.

Therefore, after following the standard software installation procedure, you bring the system up in restricted mode by turning the keyswitch to REMOTE. After creating the default set of files through the workstation Editor or by executing mBTOS Config, you bring the system up in normal mode by turning the keyswitch to MANUAL or NORMAL, as appropriate.

Note: Because having the keyswitch at MANUAL enables the **Reset** button, which could be pressed accidentally during system operation, it is recommended that you use the NORMAL keyswitch position.

Using the Normal Mode

This guide defines the normal mode to be the state of the system when it is booted up using the default set of system files.

When the system is booted up in normal mode, you have use of the complete system, defined by the files installed through the installation procedure and the system configuration files created or modified through the mBTOS Config utility.

If the keyswitch is turned to MANUAL or NORMAL and customized system files have been created for that keyswitch position (that is, files including the suffix .m or .n, respectively), the customized files are used in place of the default files. Normal mode is not used. Refer to Using Customized Modes.

Using the Restricted Mode

The restricted mode is used:

- during the initial system software installation
- to guarantee a bootable environment in case of system software or file corruption

  You should not use the restricted mode, for example, when changing .cnf files. You should reserve that mode to ensure that, no matter what changes you make, you will be able to reinitialize your system.

  For information about using restricted mode to correct corrupted system files, refer to section 12.

Capabilities of the Restricted Mode

Once the standard system software is installed and configured, the differences between the restricted and normal modes lie in how the system configuration is defined and what versions of BTOS run on the processors. Otherwise, while in the restricted mode, you access the same BTOS software and commands available in normal mode.

The restricted mode set of system configuration files (that is, the set of system configuration files that include .r) is listed in the *XE500 BTOS Installation and Implementation Guide*. They define the following configuration:

- one FP board (FP00)
- one DP board (DP00)
- one TP board (TP00)
- one CP board (CP00)
- one SP board (SP00)

This configuration means that, in restricted mode, only the first board of each type will boot up. Therefore, the workstation you work at while in restricted mode must be connected to CP00. System services reside as follows:

- The CP has the AdminAgent.
- If the master is an FP, it will have the QIC Server and the DP will have the CLI.
- If the master is a DP, it will have the QIC Server or Tape Server and the FP will have the CLI.

Error Logging

While in the restricted mode, all system errors and software installation status messages are logged in the system log file, [sys]<sys>log.sys. This is the same log file used during normal mode. You can list these messages with the PLOG command, discussed in section 12.

Using Customized Modes

This guide defines a customized mode to be the state of the system when it is booted up using a set of system configuration files and processor BTOS versions that include .m or .n.

If the keyswitch is turned to MANUAL or NORMAL and customized system files have been created for that keyswitch position (that is, the file names include .m or .n, respectively), the system is booted with a configuration defined by those files.

You cannot create customized sets of system configuration files through the mBTOS Config utility. You create them by copying existing system configuration files and including the .m or .n suffix in the new file names. You can then modify the contents of these files using the Editor.

You can modify processor operating systems with the Customizer. Refer to the *BTOS II Customizer Programming Guide*.

Analyzing and Recovering from Error Conditions

This section provides an overview of how to interpret system status and recover from errors and failures in the XE520 environment. The section contains three parts:

- Troubleshooting Tools discusses the tools that report status information and that you use to analyze problems.
- Analyzing System Status organizes system problems by functional groups. It includes information to help you analyze problems and suggests how to recover from system errors and failures.
- Recovery Tools discusses the tools and procedures that help you recover from system errors and failures.

Troubleshooting Tools

The following troubleshooting tools can help you recognize system status, errors, and failures:

- The front panel STATUS display reports the status of system start–up operations.
- BTOS status codes are numerical codes reported by the system software through the BTOS Executive and other facilities. These codes indicate the status of system operations. For the meanings of the codes, refer to your system status codes documentation. Also, through the MPERC command, you can have short descriptions of some XEBTOS status codes display on screen.
- System crash status words normally display on screen and are entered in the system log after a system crash. They can be useful in isolating the cause of a crash.
- The system log provides a detailed account of system status, errors, and failures. You can list this log by using the PLOG command.
- Installation logs provide a record of software installation operations.

- A crash dump is a copy of the memory image of a processor at the time of a nonrecoverable failure (crash). Unisys personnel can use this file to analyze the cause of a processor failure.
- The processor board's rear panel LEDs display hardware errors during system start–up and software errors during normal system operation.

Front Panel STATUS Display

The status of software installation and system start–up operations is reported through the XE520 base enclosure's front panel STATUS display. The STATUS display shows a series of two–character codes indicating the progress of the boot–up procedure. These codes are defined in appendix A.

If the unit is operating correctly, the STATUS display code sequence stops at 20, indicating that the booting procedure is complete and the system is ready to run. If codes above 20 display, a failure has prevented the system from booting up properly.

Appendix A defines display codes from 21 to 38.

The boot–up process normally takes about a minute. While waiting for a bootable disk to become ready, the STATUS display alternates between A1 and 04.

Display codes 39 through 45 indicate that a failure has occurred during a processor booting operation. The code indicates the type of processor that failed. To further isolate the type of failure associated with these STATUS display error codes, you should observe the processor LEDs on the processor board's rear panel. The interpretation of these codes is described under Status Codes on Processor LEDs.

If code DE alternates with 41, 42, 43, 44, or 45, the master processor has not received a response from processors in expansion enclosures.

BTOS Status Codes

BTOS status codes are numerical codes reported by the system software to indicate the status of system operations. Refer to your system status codes documentation for the meanings of the codes. Status codes can be reported to:

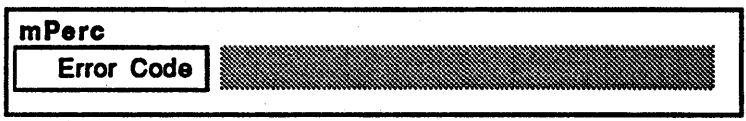
- the workstation screen through the BTOS Executive
- the system log
- software installation logs
- processor LEDs

The Executive displays error and status messages below the field displaying the last executed command in the command area of your screen. These messages reflect an activity being performed (such as Copying file XXX to file YYY) or an error that has occurred. Operator mistakes are a common source of error, and you can correct these without assistance. Common operator errors and procedures for recovering from them are discussed in the *XE500 BTOS Operations Guide.*

You can have a short description of a BTOS error code displayed at the workstation by using the MPERC command.

To execute the MPERC command, you enter **MPERC** at the Executive command line and press RETURN. Figure 12–1 displays the MPERC command form. The command form has one required field, **Error Code**, where you enter the BTOS status code number. You press GO, and a short description of the status code displays.

Figure 12–1 **MPERC Command Form**



System Crash Status Words

System crash status words can help you isolate the cause of a system crash. They normally display on screen and are entered in the system log after a system crash.

The words are numbered 1 through 8, and their values are in hexadecimal. Depending on the corresponding BTOS error code, certain status words are either unused or dependent on the BTOS error code. Refer to appendix A for a list of the status words and their meanings.

The contents of the general status register (GSR) can also help you isolate the cause of a system crash. The contents of the GSR are reported to status word 3 when a system crash occurs. Refer to appendix A for a list of the bits in the GSR and their meanings.

As an example, suppose the following error code and status words are reported as a result of a system crash:

SYSTEM CRASH - (ERC = 22)
Crash Information: 0016h 0008h D543h 0000h 0075h 0000h 0301h 0009h

The BTOS status code indicates ERC 22, a bus time-out. The information contained in the status words are as follows:

- Word 1 shows 16h, the equivalent of decimal 22.
- Word 2 indicates that the failure occurred in process 8.
- Bit 10 of word 3 indicates a bus time-out.

  Bit 6 indicates that the trapped address was generated by the central processing unit (CPU) local to the processor where the error was detected.

  Bits 1 and 0 indicate that the trapped address was greater than or equal to C0000h.
- Word 5 indicates that the processor in slot 75 was being referenced.
- Word 6 indicates that a memory address lower than FFFFh (64 kB) was being referenced.
- Words 7 and 8 indicate that the instruction preceding the instruction at address 03010:0009 was being executed when the time-out occurred.

The possible causes for the error could be isolated to a hardware failure of the processor in slot 74 or 75, or to a programming error.

The System Log

The system log, [sys]<sys>log.sys, provides a detailed account of system errors and failures. You can list it by executing the PLOG command. The PLOG command provides a listing of operation status and error reports. Each instance of certain operations and of all system errors has a separate report. The form of the report depends on the type of the corresponding operation or error. The types of reports include:

- system crashes
- bootup (system start–up) status and errors
- system initialization status and errors
- disk I/O errors
- cluster communication errors
- ISAM errors
- tape operation status and errors

Listing the System Log

To obtain a listing of the system log, you use the PLOG command. The PLOG command form allows you to exercise the following options:

- You can specify that only errors of a particular type be listed.
- You can specify a file to copy the listing to or have the listing printed automatically.
- You can specify a log file on a volume other than [sys] to take the log file information from.
- You can list only those error reports that occurred after a specified date and time.

To execute the PLOG command without exercising any of the options, you enter **PLOG** at the Executive command line and press GO. The error reports in the log file display on screen.

To execute the command with one or more options, you enter **PLOG** at the Executive command line and press RETURN. The system displays the PLOG command form shown in figure 12–2. Table 12–1 lists the fields in the PLOG command form. When you have filled in the appropriate fields of the command form, you press GO. The error reports in the log file display on screen as specified in the command form.

Figure 12–2 **PLOG Command Form**

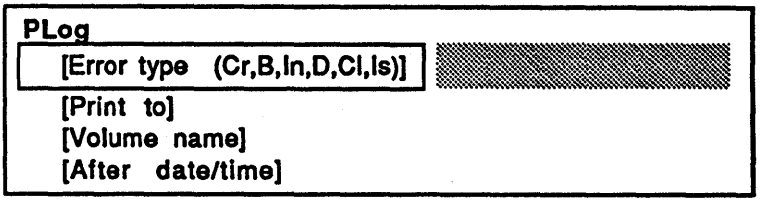


Table 12–1 **PLOG Command Form Fields**

Field	Action/Explanation
[Error type (Cr,B,In,D,Cl,Is)]	Enter the code corresponding to the type of errors you want to list. Enter only one of these codes: Cr = System crash B = System boot In = System initialization errors D = Disk errors Cl = Cluster communication errors Is = ISAM errors The default is to report all error types.
[Print to]	To create a print file of the mPLog report, enter a file name. You can then view this file on your screen or print it. To automatically send the listing to a printer and not save the print file; enter a print queue name, enclosed in square brackets. By default, the listing only displays on the screen.

Table 12–1 **PLOG Command Form Fields** (continued)

Field	Action/Explanation
[Volume name]	Enter the name of the volume you want the log file information taken from. The default is [sys]. The default directory and file name is <sys>log.sys. A log file is maintained by the operating system only on the [sys] volume, and only if it was specified in the mlVolume utility when the [sys] volume was initialized.
[After date/time]	To list only errors that occurred after a specific date and time, enter that date and time here. If a time is not specified, the time is assumed to be the start of the day.

System Crash Reports

Sample system crash error reports are shown in figure 12–3. This error report can contain the following information:

- the name of the volume the log file in
- the date and time of the failure
- the BTOS status code related to the failure
- a short description of the type of failure
- the memory location of the instruction immediately following the instruction being processed when the failure occurred
- eight hexadecimal system crash status words related to the failure

  You can use these status words to isolate the cause of the failure. An explanation of how to interpret the status words is in System Crash Status Words.
- information about the processor where a failing operation was executed

  This information can include the processor or workstation type, the memory size, the user signed on to the workstation, and the operating system the processor or workstation was booted from.

Figure 12–3 **Sample PLog System Crash Reports**

```
File Processor in slot 71H
Memory Size: 768K,
SYSTEM CRASH - (ERC = 25)                    Wed Oct 20, 1988
4:48 AM
Description: Unknown non-maskable interrupt detected
Executing instruction preceding location 03B5:009C
Crash Information: 0019H 0022H 7F98H 0000H 0071H 0000H 03B5H
009CH
Os Booted: FpBtos-B5.01/USA

Cluster Workstation, With File System
Memory Size: 960K, SignOn User Name: toni
SYSTEM CRASH - (ERC = 35)                    Thu Oct 20, 1988
PM
Executing instruction preceding location 0003:B218
Crash Information: 0023H 0008H 0000H 0000H 0000H 0000H 0003H
B128H
Os Booted: BawsClstrLfsSp-7.0.4
```

System Boot Reports

Sample system boot (start–up) reports are shown in figure 12–4. This report identifies the following information:

- the type of XE520 processor or workstation that was booted
- the slot location of the processor
- the memory size of the processor or workstation
- the date and time that the processor or workstation was booted
- the operating system the processor or workstation was booted from

Figure 12–4 **Sample PLog System Boot Reports**

```
B26, Cluster Workstation, With File System
Memory Size: 512K, SignOn User Name:
SYSTEM BOOT-                                  Fri Oct 21, 1988  4:20
 PM
Os Booted: s240ClstrLfsSp-7.0.4

Storage Processor with Storage Controller (DP) in slot 62H
Memory Size: 768K,
SYSTEM BOOT-                                  Fri Oct 21, 1988  4:20
 PM
Os Booted: DpBtos-B5.01.00/U.S.A.

Cluster Processor in slot 73H
Memory Size: 768K,
SYSTEM BOOT-                                  Fri Oct 21, 1988  4:20
 PM
Os Booted: CpBtos-B5.01.00/U.S.A.

File Processor in slot 71H
Memory Size: 768K,
SYSTEM BOOT-                                  Fri Oct 21, 1988  4:20
 PM
Os Booted: FpBtos-B5.01.00/U.S.A.
```

System Initialization Error Reports

A sample system initialization error report is shown in figure 12–5. This error report can contain the following information:

- the name and memory size of the XE520 processor that detected the initialization failure
- the date and time of the failure
- the initialization status (this is technical information useful to Unisys engineers)

Figure 12–5 **Sample PLog System Initialization Error Report**

```
File  Processor in slot 71H
Memory Size: 768K,
SYSTEM INITIALIZATION ERROR -                    Thu Oct 20, 1988
4:48 AM

Initialization Status:  2000H
```

Disk I/O Error Reports

A sample disk I/O error report is shown in figure 12–6. This error report can contain the following information:

- the name, memory size, and slot location of the processor that controls the disk with the I/O error
- the unit number of the disk
- the name of the disk's volume
- the BTOS status code related to the error
- a short description of the type of failure
- the date and time of the failure
- the number of times the I/O operation was retried before the error was deemed unrecoverable
- the disk location that was being written to or read when the I/O error was detected

  This location often indicates a previously unidentified bad spot on the disk.
- technical hardware information (in the command, main status, error status, and miscellaneous status fields) useful to Unisys engineers

Figure 12–6 **Sample PLog Disk I/O Error Report**

```
File  Processor in slot 71H
Memory Size: 768K,
DISK ERROR - XE500 Winchester Unit 3 (ERC = 305)  Wed Oct 20,
1988 3:33 PM
Description: Sector field not found (bad Cmd or bad track)
Number of Retries: 0 (Unable to Recover), Volume Name: c60345
Cylinder: 322,  Head:  0,  Sector:  0,  Number of Sectors: 0
Command: 70  38  01  42  00  00  00
Main Status:  51    Error Status:  10    Misc Status:  FC
```

Cluster Communication Error Reports

A sample cluster communications error report is shown in figure 12–7. This error report can contain the following information:

- the name, memory size, and slot location of the CP that detected the failure in cluster communications
- the BTOS status code related to the error
- a short description of the type of failure
- the date and time of the failure
- the CP's line number and cluster terminal identification number for which the error was detected
- the user signed on to the cluster terminal
- technical hardware information (in the secondary status field) useful to Unisys engineers

Figure 12–7 **Sample PLog Cluster Communications Error Report**

```
Cluster  Processor in slot 73H
Memory Size: 768K,
XE500 CLUSTER ERROR - (ERC = 8100)        Fri Oct 21, 1988
5:42 PM
Description: Cluster workstation timed out
Line: 0,  ID: 2
Secondary Status: 01  04  04  02  91  SignOn User Name:  kitsel
```

ISAM Error Reports

An Indexed Sequential Access Method (ISAM) error report includes a BTOS ISAM error code. This error code indicates the type of problem that occurred while ISAM was running. Refer to the your ISAM operations documentation for information about codes specific to ISAM.

Tape Operations Status and Error Reports

A tape status report indicates the installation status of a tape server. It can contain the following information:

- the name, memory size, and slot location of the processor where the tape server was installed
- the date and time of the installation
- the size, in bytes, of the tape server's tape buffer
- the number of tape buffers allocated for the tape server

A sample tape operation status report is shown in figure 12–8.

Figure 12–8 **Sample PLog Tape Operations Status Report**

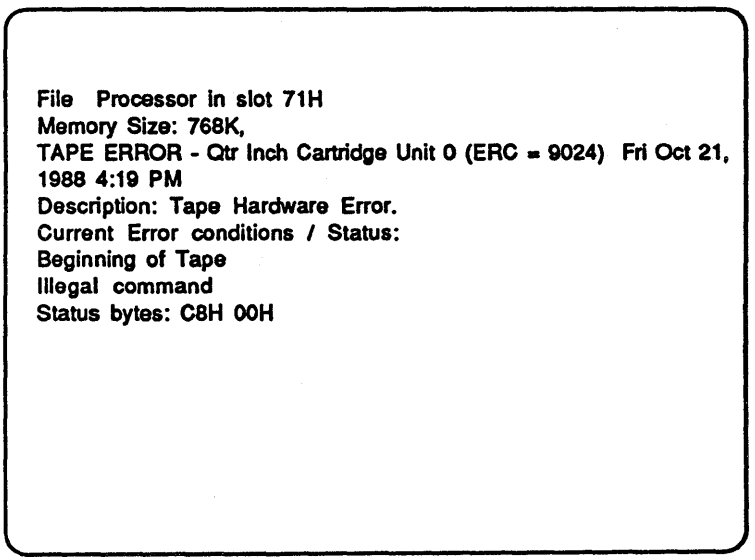
```
Storage  Processor with Storage Controller (DO) in slot 62H
Memory Size: 768K,
SUCCESSFUL TAPE SERVER INSTALLATION      Fri Oct 21, 1988
4:19 PM
Size in bytes of each Tape Buffer: 65534
Number of Tape Buffers allocated:  2
```

Tape error reports indicate errors that occur during tape operations. They can contain the following information:

- the processor the affected tape server is running on
- the half-inch or QIC tape drive unit number
- the applicable BTOS tape error code
- the current status of the tape drive, such as the current tape position

A sample tape operation error report is shown in figure 12-9.

Figure 12-9 **Sample Plog Tape Operations Error Report**



Crash Dumps

A system crash always writes a crash message to the system log file. The keyswitch position when a system crash occurs determines other system actions. If the keyswitch is set to NORMAL when a fatal error occurs, the default system action is to dump as much as possible of the failing processor's current memory image to a crash file. This file exists only if you create it. If you turn the keyswitch to MANUAL, you disable the automatic crash dump.

You can analyze the crash dump file to determine the cause of a software problem. The crash dump does not print; it is a core dump written to disk. You can use the DUMP command to get a printable hex listing of the processor's memory. The Dump utility can also compare two dump files and generate a listing of their differences.

Note: The Dump utility lists the contents of a dump file; it does not create a dump file.

For example, a dump of a suspected corrupted processor operating system can be compared to that of a known good one; you identify the corrupted areas as those that are different from the good system's.

The dump file for each processor has a unique name. For the master processor, it is [sys]<sys>crashdump.sys. The naming convention for all other crash files is

[sys]<sys>xpyy.crash

where xp is the processor type (FP, CP, SP, TP, or DP) and yy is the processor number.

The crash file for the master processor can be created when the BTOS system disk is initialized. To create the crash file for the master processor, you must enter a value for the size of the crash file in the **[Crash file]** field of the MIVOLUME command form. The size of the crash dump file should be at least equal to the size of the master processor's memory.

You must manually create all crash files other than crashdump.sys in the [sys]<sys> directory. The file size must be at least equal to the total memory available to the processor. For example, if CP00 has 768 kB of memory, you make [sys]<sys>Cp00.crash a 768-kB file.

Status Codes on Processor LEDs

If a processor detects a failure in the system start–up operations or crashes due the detection of a nonrecoverable error, the appropriate error code displays at the processor board's rear panel LEDs. Note that this processor is not necessarily the one that caused the failure. If the master processor is the affected board and the base enclosure keyswitch is at NORMAL or REMOTE, the master processor will attempt to reboot the system instead of displaying the crash code.

When a processor is booted and operating normally, LED 0 (the heartbeat LED) blinks steadily.

To determine whether the LEDs represent a hardware boot status code or a software crash code, you observe the LED lights. If the LED pattern is constant, the code displayed is a hardware boot ROM status code. If the LEDs display a walking pattern cycled with other code displays, the codes displayed between the walking pattern sequences make up a software crash hex code. For either the hardware or software crash codes, LED 0 (the heartbeat LED) does not pulse off and on.

Waiting to Boot

Once a nonmaster processor successfully completes its onboard ROM tests, it waits to be booted by the master or reset. This state is indicated on the LEDs by a walking pattern.

For processors not connected to another board, this pattern is one LED on, walking through the others.

For processors attached to an ME board, the pattern is two nonadjacent LEDs on, walking through the others.

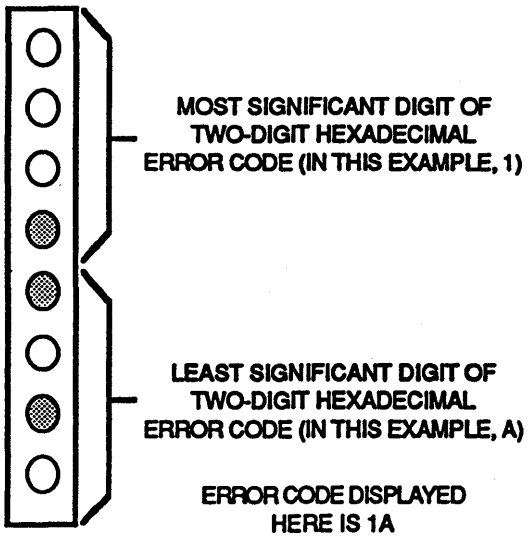
For an SP attached to an SC board to form a DP, the pattern is two adjacent LEDs on, walking through the others.

For an SP attached to an SC board and an ME board, the pattern is three adjacent LEDs on, walking through the others.

Boot ROM Status Codes

Boot ROM status codes display at the master processor's LEDs. These codes are listed in a table in appendix A. Figure 12–10 shows how to derive the two–character hex code from the LEDs.

Figure 12-10 **Reading Boot ROM Error Codes on Master Processor LEDs**



Crash Codes for BTOS Processors

After a processor crash, the affected processor enters a loop that repeats a crash display cycle on its LEDs. LED 0 (the heartbeat LED) no longer pulses. The processor stays in this loop until the system is reset.

The crash display cycle begins with four cycles of a single LED off, walking through the others.

The four walking cycles are followed by four error code display cycles. Each error code cycle displays one nibble (four bits) of the error code. This nibble represents one hex digit of the four–digit code.

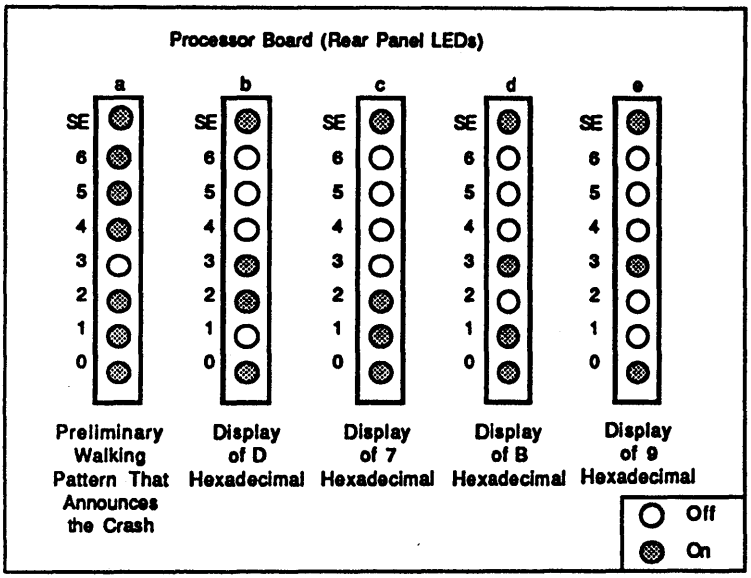
During an error code display cycle, the top four LEDs always display binary 1000 (that is, the SE, or system error, LED is on, and LEDs 4, 5, and 6 are off).

The bottom four LEDs display the four hex codes, in sequence. LED 0 is the least significant bit of the nibble value. The first nibble displayed is the most significant nibble of the hex error code. At the end of each nibble display, all LEDs blink off to indicate that the next nibble will follow. After the last nibble displays, the entire crash display cycle repeats. For example, if a processor displays, in order, D, 7, B, and 9, the error code is hex D7B9. Figure 12-11 shows a sample sequence of this crash code display.

To record a crash error code, you wait for the beginning of the crash display cycle (the four walking cycles). Then you record the value of each nibble as it displays. You repeat the procedure to verify that you have recorded the error code properly.

You must convert the code to decimal. You can then look up its meaning in your system status codes documentation.

Figure 12–11 **Sample Processor LED Crash Code Sequence**



Analyzing System Status

This subsection organizes system problems by functional groups. It includes information for problem analysis and suggestions for recovery from system errors or failures.

System Start–Up

To diagnose system start–up problems, it is helpful to know the start–up process. The next subsection discusses the operations that occur at system start–up.

What Happens during System Start–up

When powered on, the XE520's processors perform a series of self–tests. These tests are controlled by instructions in their read–only memory (ROM). The tests include:

- ROM readability
- memory initialization for error correcting code (ECC)

- memory test of all accessible random access memory (RAM), including RAM on Memory Expansion (ME) boards
- I/O tests on remote reference registers

In addition to these tests, the master processor (the FP or DP in the first processor slot of the base enclosure) performs an extensive set of tests to verify that the I/O paths to the disks it controls are working properly. All other processors:

- program their interrupt systems
- set up a stack in high memory
- unmask the doorbell interrupt (Int 1)
- mark a flag in the CPU description table that determines whether the processor will perform a core dump to a crash file in the event of a detected unrecoverable error
- enter an enabled loop waiting to have their operating systems sent by the master processor

When the master processor has completed its self–tests, the booting process begins.

The master processor reports the status of the start–up procedure by displaying codes at the front panel STATUS display. If a hardware failure occurs, the appropriate codes display in the STATUS display and on the processor LEDs of the processor that detected the failure.

The master processor looks for its operating system, [sys]<sys>SysImage.sys, sequentially through the disk units in the system: first d1, then d2, and so on.

When found, [sys]<sys>SysImage.sys is copied into the master processor's memory. The master processor reads its configuration file to establish its disk I/O device parameters. Its initialization file is then executed to install the appropriate system services.

This master operating system then takes over the task of loading the rest of the processors with their operating systems. The master operating system reads the master configuration file, [sys]<sys>Master.cnf, to determine which processors should be booted, the operating systems they should be booted with, and, for some processors, their configuration files.

As each processor is booted, it reads its configuration file to establish its I/O device parameters. Its initialization file is then executed to install the appropriate system services.

When all processors have been booted and their system services installed, the system enters normal operation.

System Start–Up Problems

The master processor or the system disk is the most likely cause of most start–up problems.

ERC 301s are frequently logged in connection with boot problems. They represent disk I/O errors, which are caused by an unreadable disk (bad spots), unstable input power, or improper drive termination.

Problem: Master processor freezes while attempting to load operating system to a processor. The master processor LEDs display an error code, but no error is entered in the system log. Some or all of the other processors have not booted (that is, one LED on cycles through the processor LEDs).

Possible causes:

- The master processor is faulty.
- Two processor boards are jumpered together by a processor–to–ME board interconnect bus.
- The backplane was damaged during the recent installation of a board.
- The power supply is malfunctioning.

Recommended action: Contact your Unisys field service representative.

Problem: System fails to boot; 21 appears in the STATUS display.

Possible causes:

- The master processor is faulty.
- Disk drive cables (data or control) are faulty.
- The disk on which the master operating system ([sys]<sys>SysImage.sys) is stored is faulty.

- Another processor in the base enclosure is faulty.
- For a base enclosure with a QIC tape drive, the first built–in disk drive is not properly terminated.
- The master operating system is corrupted.

Recommended action: 21 in the STATUS display means no valid master operating system can be found on the first disk recognized to have space allocated for the file [sys]<sys>SysImage.sys. This allocation happens when the disk is initialized.

If you suspect that the BTOS file system is corrupted, refer to the recovery procedure under Determining Corrupted System Files.

If No such device (ERC 215) is returned when looking at the system disk, a fundamental hardware problem exists between the master processor and the system disk. Contact your Unisys field service representative.

Problem: System freezes during start–up; 50 appears on the STATUS display.

Possible causes:

- There is a syntax error in a processor initialization file, making the run statements in the file unexecutable.
- The BTOS file system is partially corrupted.

Recommended action: Boot the system in restricted mode and verify that the processor initialization files are correct. If they are in order, use the recovery procedure under Determining Corrupted System Files.

Problem: A processor board crashes during system boot; the board flashes a crash code on its rear panel LEDs.

Possible causes:

- System files are corrupted.
- The system disk is faulty.
- The processor board that crashed is faulty.

Recommended action: If you suspect corrupted system files, use the recovery procedure under Determining Corrupted System Files. Check the operating system file that is loaded into the crashed processor.

Problem: Code DE alternates with code 41, 42, 43, 44, or 45.

Possible causes:

- An expansion enclosure is not turned on.
- A bus repeater board or the cabling between the bus repeater boards is faulty.

Recommended action: Make sure all expansion enclosures are powered on.

If all expansion enclosures are powered on and code DE still displays, contact your Unisys field service representative.

Problem: System is unusually slow.

Possible causes:

- The system disk is fragmented.
- Memory expansion is not recognized by the system.
- The system disk is busy with recoverable read errors.
- Not enough free blocks exist on the disk for context file use (disk full).
- X, Y, or Z blocks require better allocation to match processor I/O demands.
- A customization of BTOS processor operating systems is required to better allocate system resources.

Recommended action: Perform a backup, reinitialization, and restore of all disks believed to be fragmented (refer to section 3). If a disk is full, move or delete files from it.

Check the system log, using the PLOG command, to see that all processors with Memory Expansion (ME) boards have the proper amount of memory. If they do not, contact your Unisys field service representative.

Use the procedures in section 10 to determine the I/O communication performance of the processors. Adjust X, Y, or Z block allocation accordingly.

Common User and System Errors

The system reports both status messages and BTOS status codes through the Executive, where they display on screen.

Your *XE500 BTOS Operations Guide* discusses general status messages.

The most common errors that result from user mistakes are file, directory, and volume management errors:

- invalid file specifications (for example, a user provides an invalid file name or requests an operation on a file that does not exist)
- security violations (for example, a user attempts to modify a read–only file or enter an invalid password)
- overflow conditions (for example, a directory or a volume fills up)

To recover from this type of error, you need only correct the entry that caused the error.

Other file errors result from disk malfunctions that have caused the loss of volume control structures. You can recover from damage to volume control structures caused by disk malfunctions by backing up, reinitializing, and restoring each volume.

Certain device errors resulting from user oversights are easily recoverable. Such errors include improper initialization of the device, unit power being off, or the medium for a device not being on line (for example, a tape is not properly loaded into the tape drive).

Printer spooler errors include configuration errors (such as attempting to add a printer to a channel that is not free) and incorrect security procedures (for example, specifying a password when one is not required).

Intermittent System Crashes

The master processor monitors all processors within the XE520 system by sampling internal locations (ordinarily active) in other processors. If a processor continues to be inactive, the error is logged, and the whole system is brought down to prevent damage. The front panel STATUS display at the XE520 base enclosure indicates the type of failing processor.

If the problem is due to software, rebooting will normally allow you to return the system to operational status. If the problem is a hardware malfunction of one of the processor boards, you can remove that processor's entry in the master configuration file and reboot the system to run without the failed processor. For information about the master configuration file, refer to your installation and implementation documentation.

Causes of intermittent crashing of the system can range from bad input power to a mix of software revisions running concurrently. Analyzing system log entries and crash dump files associated with each crash are starting points to determining the source of the problem. The following is a list of typical crashes, with possible causes and recommended actions.

Problem: ERC 22 (bus time–out), ERC 28 (Bus error).

Possible causes:

- A processor board is poorly seated with the backplane.
- Hardware is not distributed across the system bus properly.
- The crashing processor or the processor involved in a remote reference is faulty.
- An attempt was made to access a nonexistent processor.

Recommendation action: XE520 processors can assume two roles, that of a bus master or that of a bus subordinate. A processor is a bus master when it writes and reads another processor's memory, a bus subordinate when its memory is written and read by the bus master. A bus time–out (ERC 22) occurs on a bus master if a bus subordinate does not acknowledge a request in 15 microseconds.

A subordinate processor experiencing a crash such as an ERC 21, ERC 23, and so on, issues the bus error signal to its master. The master then crashes with an ERC 28.

Refer to the system log, using the PLOG command, to identify the two boards involved. The board on which the ERC 22 or ERC 28 occurs is always the master; the subordinate is identified by the slot number in system crash status word 5 in the error report associated with the crash.

Problem: System freezes (no error logging).

Possible cause: A power supply of less than the minimum revision is being used. These supplies are at times unable to source the high current demand required by a fully populated enclosure.

Recommended action: Contact your Unisys field service representative.

RS–232–C Serial Communications Problems

Problem: Corrupted or no communications between a CP or TP and an RS–232–C serial device.

Possible causes:

- There is no entry for the RS–232–C channel in the processor's configuration file or the entry is not properly defined.
- The RS–232–C port on the processor is faulty.
- The cable is faulty or the wrong cable is being used.

Recommended action: Check that the entry for the channel in the processor's configuration file exists and is correct.

Cluster Communications Problems

Problem: Corrupted or no communications between a CP and a cluster terminal. Error codes 8100 (time–outs) or 8103 (protocol failures) are reported. B25 series workstations cannot run at the high baud rate of 1.8 Mbps.

Possible causes:

- The cluster lines are not properly terminated.
- The RS–422 port on the processor is faulty.
- The cable is faulty.

Recommended action: The total length from the beginning of an RS–422 channel on the CP board to the end of the cluster line can be up to 1200 feet at baud rates of either 307 Kbps or 1.8 Mbps. This 1200–foot maximum length is a total of all cable lengths between cluster terminals and consequently between cluster line terminators.

Make sure that both lines associated with a cluster port are properly terminated using an RS–422 100–ohm terminator.

Check that all RS–422 cables are properly connected and not defective.

Disk Drive Problems

Problem: Error 301 (Disk I/O error) is reported.

Possible causes:

- Previously unrecorded bad spots appear on the disk.
- The disk is deteriorating, producing unwriteable or unreadable areas.
- The AC and DC grounds are connected.
- Input power is unstable or inadequate.
- The disk drive daisy chain is not terminated properly (for example, there is no terminator or there are multiple terminators in the chain).

Recommended action: Perform a backup, reinitialization, and restore of the disk.

If ERC 301 errors continue being reported, contact your Unisys field service representative.

Problem: BTOS cannot access disk drive.

Possible causes:

- The configuration file for the FP or DP is missing, incorrect, or not included in the master configuration file.
- The disk drive configuration file for the disk type does not exist or is incorrect.
- The FP or DP is faulty.
- Disk drive cables are faulty.
- The disk drive's power supply is malfunctioning.
- For an SMD in an MD3 enclosure, the chassis of the SMD drive is not grounded to the XE520.

Recommended action: Make sure that the configuration file for the FP or DP exists, is correct, and is included in the master configuration file for all FPs and DPs except the master FP or DP. Also make sure that the disk drive configuration files exist and are correct. For more information about these configuration files, refer to your installation and implementation documentation.

Recovering from System Problems

The following subsections outline procedures for recovering from some system problems.

If corrupted system files have caused a system failure, whether due to a faulty system disk or to missing or incorrect system files, the recovery approach is to boot the system from another set of system files. This set can be restricted mode system files if the system disk is operating properly.

If system files become corrupted, certain system services fail or the system does not boot. For example, if a master BTOS command run file is corrupted, the command fails when you attempt to execute it. Or, if the master operating system, [sys]<sys>SysImage.sys, is corrupted, the system is unable to boot, even in the restricted mode.

If only certain system services fail and you suspect corrupted system files, you should

- boot up in restricted mode
- if you know which files are corrupted, use the mRestore utility to restore those files from your system software archive media
- if you are not sure which files are corrupted, reinstall the software package that contains the files you suspect are corrupted

If the system fails to boot in either the restricted or the normal mode, you should reinstall restricted mode software. Then you should see if the corrupted system files can be corrected, or use the mRestore utility to restore system files.

If the system fails to boot again, the system disk probably has a hardware failure. Reinstall restricted mode software to a disk other than the faulty system disk.

Using the Restricted Mode

If the system disk is not faulty or bad spots have only corrupted normal system files, you should be able to bring the system up in restricted mode. The capabilities of restricted mode are discussed in section 11.

Determining Corrupted System Files

To determine which system files are corrupted, use the following procedure:

1 Boot the system in restricted mode.

2 Use the Editor to review the system configuration files on the system disk (that is, the master configuration file, processor initialization files, processor configuration files, and so on).

If the system configuration files are present and correct, go to step 3.

3 Use the COPY command to copy the entire system disk to [nul].

Enter **[sysdisk]<*>*** in the **File from** field, where sysdisk is the device or volume name of the system disk.

Enter **[nul]<*>*** in the **File to** field.

4 If ERC 301s (disk I/O errors) appear during the copy operation, take note of the files for which the ERC 301s were reported. Then use the PLOG command to check the system for the location of the unreadable area (bad spot) of the disk in cylinder, head, and sector format.

If no ERC 301s are reported, go to step 8.

5 Back up the system disk using the mBACKUP VOLUME or mTAPE BACKUP VOLUME command.

6 Reinitialize the system disk, entering new bad spots found during the Copy operation.

7 Restore the files to the system disk using the mRESTORE or mTAPE RESTORE command.

Restore the corrupted files found in step 4 from your system software archive media, or reinstall the files from your software release media.

8 Use the DUMP command to compare system files that you suspect are corrupted against the restricted mode version or the version on the bootable removable medium. Corrupted files are indicated if the Dump utility reports differences between the two versions.

You can then restore the corrupted files from your system software archive media or reinstall the files from your software release media.

Running the System in a Degraded Mode

If you suspect processor board hardware failure as the cause of system problems, you can run the system in a degraded mode.

If the failed processor is not essential for normal system operation, you can remove the processor's entry from the master configuration file, [sys]<sys>Master.cnf, and reboot the system. This allows you to run the system until the failed processor can be replaced.

Reporting Problems

You should report all software problems, software deficiencies, and suggestions for improvements by submitting a User Communication Form (UCF). Please review the UCF for completeness and forward it to your local Unisys field representative.

All information that would be helpful for understanding and analyzing the reported situation should accompany the UCF. This can be in the form of descriptions, computer listings, media containing programs and data, and so on. Please include a listing of the file [sys]<sys>Sys.Version with all UCFs submitted.

Example problem situations appear below, with appropriate information to be submitted for each.

For a hang or crash condition:

- a printout from a PLog or the system log file itself ([sys]<sys>log.sys)
- any observations of abnormal operation prior to the problem
- applicable crash files (for the master processor, [sys]<sys>crashdump.sys; for all other processors, [sys]<sys>Xpnn.crash, where Xpnn is the processor name, such as FP01 or CP00)
- if you have customized a processor's operating system, the symbol file that was created when you used the LINK command to link the operating system
- an indication of the front panel STATUS display code
- an indication of the LED lights on the back of each processor board
- a description of the operation being performed when the problem occurred
- any data or programs (source and run files) necessary to demonstrate the problem

  You should copy these to a diskette and send it along with instructions for recreating the problem.

For an execution condition:

- any data or programs (source and run files) necessary to demonstrate the problem

  You should copy these to a diskette and send it along with instructions for recreating the problem.
- a list of erroneous output with the errors indicated and corrections noted

For a problem that recurs but cannot be reproduced on demand:

- a clear and detailed description of the situation
- a listing of the system log file (where applicable)
- any data available for this type of problem, including listings

When completing the UCF, please observe the following guidelines:

- You should report only one problem per UCF.
- You should use the UCF form only for reporting system software problems, errors in documentation, or suggestions for new software features. You should not use it to ask questions or request information.
- You should make the information as clear and legible as possible.
- You should clearly mark all attachments with the UCF number. If you want anything to be returned to you, including any media you have submitted, you must indicate so on the UCF form and the media. If you do not, your media will not be returned.

Status Code Tables

Kernel

21

Memory reference fault.

System crash status word 3 is the nonmaskable interrupt (NMI) status code. Status word 4 is the high 2 bits of the memory address at the time of the NMI.

22

Bus time–out.

A bus time–out usually results from a reference to a nonexistent remote processor memory address. If system crash status word 4 is a 3, a remote reference was being performed when the crash occurred. Status word 5 is the contents of the remote slot number register (port hex 40). Word 6 is the contents of the base register zero (port hex 48).

23

Double–bit error detected by error correction code (ECC) circuitry.

System crash status word 3 is the NMI status code. Status word 4 is the high 2 bits of the memory address at the time of the NMI. Status word 6 contains the ECC syndrome register contents.

24

Power failure.

System crash status word 3 is the NMI status code. Status word 4 is the high 2 bits of the memory address at the time of the NMI.

25

Unknown NMI.

System crash status word 3 is the NMI status code. Status word 4 is the high 2 bits of the memory address at the time of the NMI.

26

Stray interrupt.

System crash status word 3 is the interrupt type multiplied by 6. Status word 4 is the interrupt service register, word 6 is the interrupt request register, and word 7 is the interrupt mask register.

The stray interrupt number referred to in the log entry for this error code is not significant; it is indicating the type of interrupt that the processor has attempted to service when no actual interrupt condition has occurred. The problem is caused by an invalid program jump to the processor's low memory interrupt vector table.

27

Divide overflow.

This is usually caused by a divide by zero. System crash status words 7 and 8 indicate the address of the instruction following the instruction being executed when the crash occurred.

28

Bus error.

This error results from a reference to a nonexistent remote processor memory address or a remote double-bit error. If system crash status word 4 is a 3, a remote reference was being performed when the crash occurred. Status word 5 is the contents of the remote slot number register (port hex 40). Word 6 is the contents of the base register zero (port hex 48).

39

Cannot resign as a server of a request you are not serving or attempt to serve a request that is already being served.

Initialization

104

Insufficient Y blocks or Z blocks specified at system generation.

Increase the number of Y blocks or Z blocks of the failing processor by either recustomizing its operating system or modifying the Y or Z block allocation entry in the processor's configuration file. Refer to the *BTOS II Customizer Programming Guide* or the *XE500 BTOS Installation and Implementation Guide* for more information about Y and Z blocks.

105

Insufficient hardware configuration detected.

There are no CPs running or there are too many FPs, DPs, or SPs running.

106

A processor initialization file contains an error.

Correct the processor initialization file for the failing processor. The *XE500 BTOS Installation and Implementation Guide* contains information about the proper format for processor initialization files.

107

Watchdog process on master processor detected failure of another processor on the bus.

This error will only be returned if a processor other than the master processor crashes and the NoWatchDog entry has been removed from the XE520 master configuration file, [sys]<sys>Master.cnf.

150

A processor attempted to route a request to itself, which cannot be done.

151

Remote request too large for local buffer.

Try increasing the number of Y or Z blocks for the failing processor. Refer to the *BTOS II Customizer Programming Guide* or the *XE500 BTOS Installation and Implementation Guide* for more information about Y and Z blocks.

152

Cannot remote boot a processor that has not been initialized.

153

Target processor did not respond to remote boot.

154

Illegal option used in GetSlotInfo or GetProcInfo, or illegal code in the Cdt issued from RemoteBoot.

155

Too many disk devices declared in the [sys]<sys>Master.cnf file.

Increase the size of %nmounted disks in the system generation prefix file table for the master processor and regenerate the master processor's operating system. For more information about customizing XE520 processor operating systems, refer to the *BTOS II Customizer Programming Guide.*

Device Management

305

Attempt to reference unformatted disk.

Any attempt to use an unformatted disk by any I/O request, except by the format request, returns this error.

306

Recall failed.

307

Hard disk controller write fault detected.

350

Hard disk controller local direct memory access (DMA) fault.

352

Hard disk controller remote DMA fault.

Allocation (BTOS)

403

Parameter in a BTOS memory management request is not aligned on a paragraph boundary (that is, the offset part of the segment:offset memory address must be zero).

Printer Spooler

709

Current print request cancelled.

710

Printer restarted.

711

Printer is freed and needs to be reconfigured.

720

Too many printers specified in spooler configuration file. The limit is 100.

721

Attempt to install a printer spooler manager on a processor that already has a printer spooler manager installed.

Check that the failing processor's initialization file has only one printer spooler manager entry.

722

Attempt to install a printer spooler manager on a processor that does not have printer hardware. It can be installed only on a CP or TP.

Check your FP, DP, or SP initialization files to make sure that they do not contain an entry for a printer spooler manager.

723

Attempt to install a printer spooler manager when there is no Queue Manager installed. (The spooler tries for two minutes, and then gives up.)

Check your processor initialization files to make sure that they contain an entry for the Queue Manager.

Tape Management (BTOS)

9002

Tape operation timed out.

This error is usually caused by an operator error or a hardware malfunction.

9009

End–of–tape encountered.

For the QIC server, this is the logical end–of–tape (in write mode, the estimate of the remaining tape space has reached zero; in read mode, blank tape has been reached).

9010

Unrecoverable I/O error occurred during a read or write operation.

9016

Tape drive is not ready.

Tape drive is either off–line or busy.

9021

A file mark was encountered during a read operation.

9024

A hardware error occurred in the tape drive or the controller.

9032

A tape service request was made with an invalid tape handle or user number.

9033

A tape service was made with invalid parameters.

9034

The half–inch tape server ran out of device control blocks.

9035

An open tape service was requested with an invalid tape name.

9036

An open tape service was requested on a drive currently being used by another user.

9038

An unrecognized command was issued as a tape operation request.

9039

A read or write request was issued to a half–inch tape drive that had tape errors outstanding.

9040

An open tape service was requested with an unrecognized open mode.

9041

A close tape service request was issued to a half–inch tape drive that had operations outstanding.

9050

During read mode, a tape was detected as being corrupted.

9051

During write mode with a QIC tape, the physical end–of–tape was reached before the logical end–of–tape; therefore, information was lost.

9052

Invalid tape position.

A half–inch tape is neither at load point nor at the start of a file.

9053

Invalid tape reel.

The tape is not part of the archive tape set currently being used for a restore operation.

9054

Invalid tape sequence.

The tape is not the next archive tape in the current sequence being used for a restore operation.

9055

Invalid tape configuration file for the BTOS tape utility.

9056

Missing tape configuration file for the BTOS tape utility.

9057

Invalid tape record size specified for buffer in tape read/write request.

9058

Missing tape controller hardware.

9070

No internal buffers are available for remote processor copy-in/copy-out. This will happen only if the read/write request is from a remote processor.

Reissue the request.

9071

Bad alignment of user-supplied data buffer.

Be sure data buffers are aligned on word boundaries.

9072

The user request specified a data buffer size larger than that specified at tape server installation time.

9073

QIC tape was removed from the drive after being opened by a user.

9074

QIC server was given an illegal command.

9075

QIC tape is not a bootable tape.

9076

QIC tape is a corrupted bootable tape.

9080

The user of tape access methods supplied a buffer that was too small. The buffer must be at least three times the record size specified in the tape configuration file used by the access method.

9081

Too many tape servers installed in the system.

9099

An attempt was made to install a tape server on a processor that already had one.

BTOS System Requests

11800

Internal waiting–on–Yblk error.

11801

Time–out waiting for Data Set Ready (DSR) after an open terminal request.

11802

The target processor does not respond.

The processor has crashed or has a hardware failure.

11803

Routing area too small.

When adding request codes, the size of the routing area in the master processor's CDT was not increased.

Increase the value of the sRouteArea parameter in the master processor's Prefix.asm file to accommodate the additional requests. Refer to the *BTOS II Customizer Programming Guide.*

11804

Illegal parity option.

A mark or space parity was specified with eight data bits; the hardware does not allow this combination.

11805

The user number table used to map global user numbers to local user numbers is too small.

Increase the value of the %nRemoteUser parameter in the SysGen.mdf file. Refer to the *BTOS II Customizer Programming Guide.*

AdminAgent

11900

The default path supplied to MfAdminAgent.run from WsAdminAgent.run is not a valid XE520 path (for example, the path is local to a workstation volume).

11901

The AdminAgent is busy and cannot service the request.

11902

The internal request block sent from the WsAdminAgent to the MfAdminAgent has an improper format.

This error can occur if the WsAdminAgent and MfAdminAgent run files are not at the same version level.

System Start–up/Initialization Sequence

0

The system is in the reset mode (keyswitch is in the STOP position).

01

Starting boot procedure; performing boot ROM diagnostics.

02

Boot ROM writing memory to [sys]<sys>crashdump.sys.

03

Boot ROM performing full memory read/write.

04

ROM booting the master OS, [sys]<sys>SysImage.sys, on the master processor. Display alternates between 04 and A1 while waiting for system disk to become ready.

05

Boot ROM sequence successfully completed.

06

BTOS initialization begun.

07

Hardware initialization complete. First time master processor communicates across system bus.

08

Master processor initialization complete.

09

Master processor initializes disk hardware (reads [sys]<sys>Fp00.cnf or [sys]<sys>Dp00.cnf).

10

Master processor reads master configuration file, [sys]<sys>Master.cnf, and initializes other processors. Processor BTOS download is in progress.

20

System is operating normally.

Boot Medium Error Codes

21

The master processor ROM was unable to find a ready disk.

23

The master processor ROM found no valid home block on the first ready disk drive.

24

The master processor ROM found a bad run file signature on the first ready disk drive.

25

The run file checksum of the master OS, [sys]<sys>SysImage.sys, on the first ready disk is bad, indicating the file is corrupted.

26

Error when trying to reset.

27

Error when trying to rewind the tape.

28

Error when reading the tape.

29

Tape was written with incorrect or unsupported blocking factor.

Disk Hardware Error Codes

30

Disk error on drive 0.

31

Disk error on drive 1.

32

Disk error on drive 2.

33

Disk error on drive 3.

36

Bad disk controller; ROM could not read the controller register.

37

Bad disk controller; not ready in command phase.

38

Bad disk controller; no response.

Processor Error Codes

39

General board hardware error. A processor has reported a hardware error; see processor's rear panel LEDs for specific error.

40

Master processor detected FP time–out, indicating processor crash.

41

Master processor detected TP time–out, indicating processor crash.

42

Master processor detected CP time–out, indicating processor crash.

43

Master processor detected SP time–out, indicating processor crash.

44

Master processor detected DP time–out, indicating processor crash.

45

Master processor detected processor time–out, indicating processor crash. Processor type cannot be determined.

Software Error Codes

50

Fatal software error detected. See processor rear panel LEDs for error code.

Expansion Enclosure Error Codes

DE

There is a break in the system bus between enclosures due to one of the following conditions: (1) An expansion enclosure is not turned on; (2) there is a faulty bus repeater board; (3) bus repeater cables are improperly installed or damaged.

Codes Displayed While STATUS Display Is at 01

01

Starting board hardware initialization.

02

Performing memory test of first 64 kB of memory.

03

Setting up initial memory area.

04

Determining memory size.

05

Relocating BTOS buffer. Setting up CDT.

06

Checking I/O ports.

07

Performing ECC wash to set up check bits.

10

Hardware initialization successfully completed.

Codes Displayed When STATUS Display Is at 39

0A

Bad ROM checksum.

0B

Bad I/O port.

0C

Memory error in local memory while writing 0s.

0D

Memory error in local memory while writing 1s.

0E

Memory error in local memory while writing address pattern.

0F

Memory error in local memory while reading address pattern.

Bad Media Error Codes Displayed When STATUS Display Is at 21

21

Cannot boot from any disk.

22

No tape or disk device contains a valid master OS, [sys]<sys>SysImage.sys.

23

No volume home block found on first ready drive.

24

Bad run file signature on first ready drive.

25

Bad run file checksum on first ready drive.

Master DP-Specific Boot Medium Error Codes

(These codes also appear on the front panel STATUS display.)

26

Error when trying to reset.

27

Error when trying to rewind the tape.

28

Error when reading the tape.

29

Tape was written with incorrect or unsupported blocking factor.

Disk Hardware Error Codes

(These codes also appear on the front panel STATUS display.)

30

Disk error on drive 0.

31

Disk error on drive 1.

32

Disk error on drive 2.

33

Disk error on drive 3.

Disk Hardware Error Codes Displayed When STATUS Display Is at 39

36

Bad disk controller; cannot read controller registers.

37

Bad disk controller; never ready in command phase.

38

Bad disk controller; no response.

Master DP–Specific Disk Hardware Error Codes Displayed When STATUS Display Is at 39

3A

Storage Processor (SP) of Master DP does not have a Storage Controller (SC) board attached.

3B

Inconsistent values on SMD controller interface.

3C

SMD controller Data Input/Output (DIO) bit set incorrectly.

3D

A fatal SMD I/O error has occurred. (This is a generic error.)

ME Board Error Codes Displayed When STATUS Display Is at 39

4C

Memory error on ME board while reading 0s.

4D

Memory error on ME board while reading 1s.

4E

Memory error on ME board while writing address pattern.

4F

Memory error on ME board while reading address pattern.

General Status Codes

1A

Error during hardware initialization.

1B

Error during data communications (data comm) peripheral initialization.

1C

Starting data comm dump or boot.

1D

Writing data comm dump.

1E

Reading data comm boot.

1F

Waiting for data comm line ACKnowledge.

81

Starting hardware initialization.

82

Starting checksum of ROM.

84

ECC logic and memory are initialized nondestructively (data is not lost).

FF

Processor is in reset state.

Table A–1 **System Crash Status Word**

Word	BTOS Error Codes	Meaning
1	All	The hexadecimal value for the corresponding BTOS decimal error code.
2	All	The process number that was running when the error occurred.
3	21, 22, 23, 24, 25, 26, 28	The 16–bit nonmaskable interrupt (NMI) status (read from the General Status Register, port hex 68; refer to table B–2).
4	21, 22, 23, 24, 25, 28	The high two bits of the memory address at the time of the NMI. If the value is 3, a remote memory reference was being performed when the crash occurred.
	26	The interrupt service register.
5	22, 28	The contents of the remote slot number register (port hex 40). This value represents the slot number of the processor being referenced when the crash occurred.
6	22	The contents of the base register zero (port hex 48). The base register zero holds bits 16 through 31 of the off–board memory address.
	23, 28	The error correction code (ECC) syndrome register contents.
	26	The interrupt request register.

Table A–1 **System Crash Status Word** (continued)

Word	BTOS Error Codes	Meaning
7	All (except 26)	The code segment (CS) of the location following the call to the BTOS fatal error handler.
	26	The interrupt mask register.
8	All	The instruction pointer (IP) of the location following the call to the BTOS fatal error handler. Status words 7 and 8 indicate the memory location of the instruction immediately following the instruction that was being executed when the error occurred. The value in the CS register, shifted left one digit, plus the value in the IP register gives the address.

Table A–2 **General Status Register Contents**

Bit	Name	Bit Setting	Meaning
15	ME	0	Memory Expansion board is installed.
14	SE	0	At least one processor in the system is driving the system error line on the system bus.
13	LR	0	Last reset by power up or RESET switch.
		1	Last reset by software.
12	NEM	0	If an NMI was caused by a reference to a nonexistent memory location, the reference could have been initiated locally, or by a processor or direct memory access (DMA) controller on another board.
11	PF	0	Power failure.
10	TO	0	Bus time–out. No acknowledge was received from a memory or I/O timing circuit.

Table A–2 **General Status Register Contents** (continued)

Bit	Name	Bit Setting	Meaning
9	BE	0	Bus error. A remote double–bit error or a reference to a nonexistent memory location occurred.
8	DBE	0	Double–bit error.
7	REM	1	The trapped address was generated by another processor.
6	CAD	1	The trapped address was generated by the local CPU.
5	DMA	1	The trapped address was generated by the local DMA controller.
4	REM	1	Front panel keyswitch is in the REMOTE position. Only applicable for the master processor.
3	DFP	1	This board is the master processor.
2	NOR	1	Front panel keyswitch is in the NORMAL position. Only applicable for the master processor.
1	A19		Bit 19 of the trapped address. Used with bit 0 (A18) to determine if trapped address is located in base memory, expansion memory, or on another processor.
0	A18p		Bit 18 of the trapped address. If bits 19 and 18 are both 0, the trapped address is located in base memory, expansion memory, or on another processor. If bits 19 and 18 are both 1, the trapped address is located on another processor. Otherwise, the trapped address is located in expansion memory.

Hardware Configuration Information

Figure B–1 **Built–in Disk Device Naming Conventions**

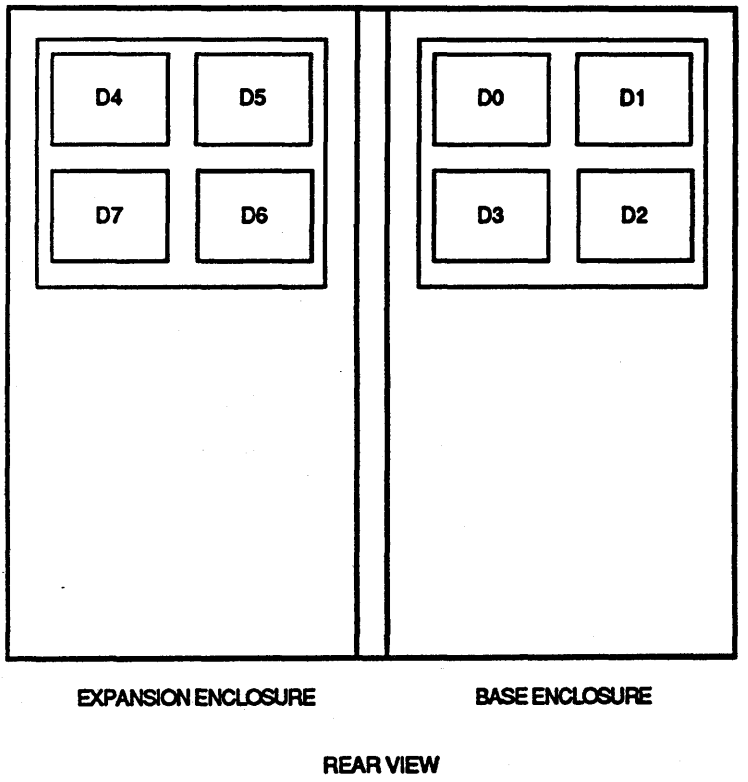


Figure B–2 **Built–in Disk Device Naming Conventions for DP00**

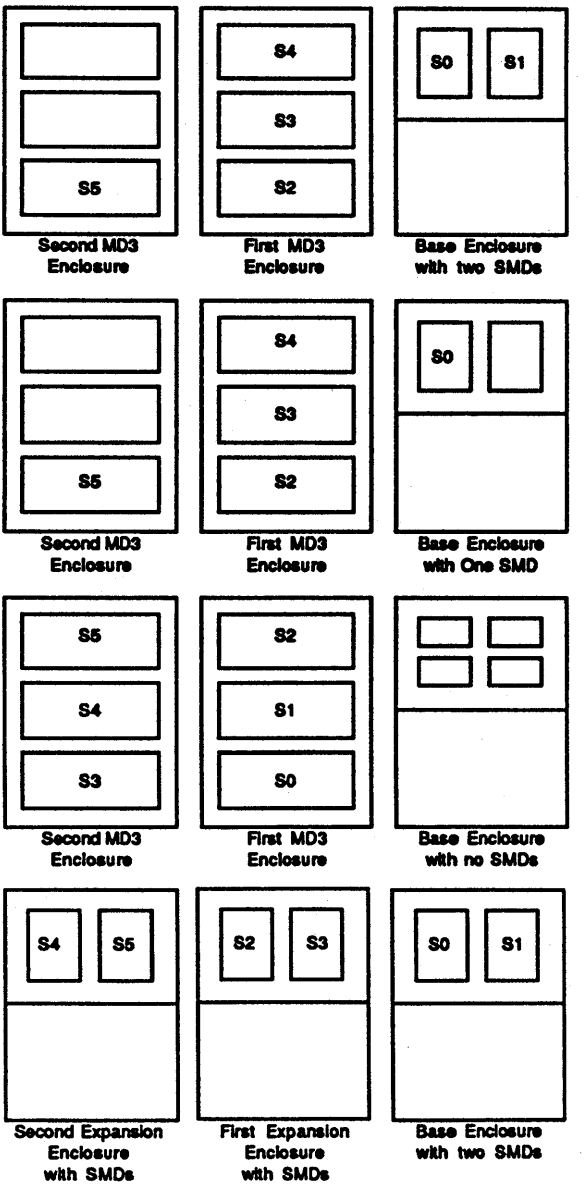


Figure B–3 **Processor Board Numbering Scheme**

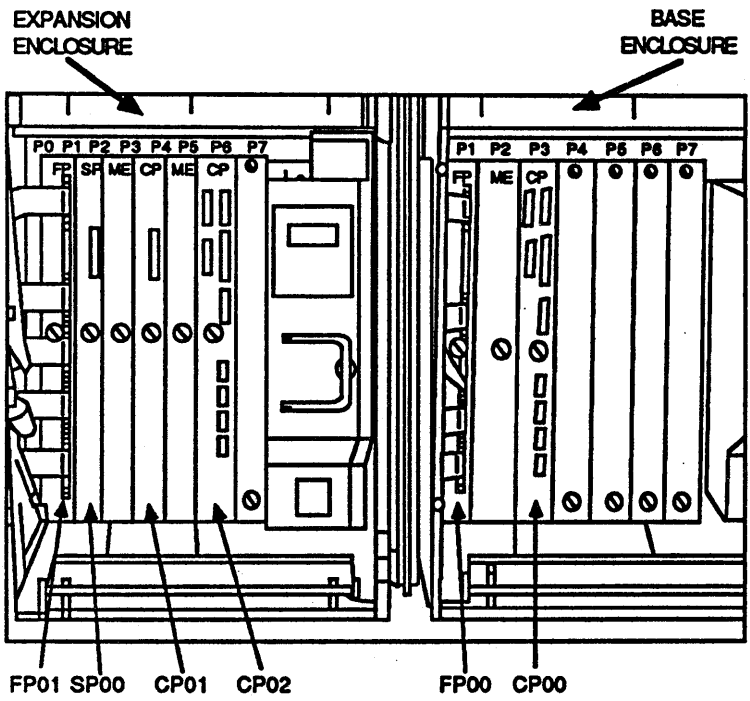


Table B–1 **Processor Slot Numbering Scheme**

Enclosure	P0	P1	P2	P3	P4	P5	P6	P7
Base	70	71	72	73	74	75	76	77
Expansion 1	60	61	62	63	64	65	66	67
Expansion 2	50	51	52	53	54	55	56	57
Expansion 3	40	41	42	43	44	45	46	47
Expansion 4	30	31	32	33	34	35	36	37
Expansion 5	20	21	22	23	24	25	26	27

Glossary

AdminAgent. AdminAgent is the system service that manages the execution of the master utilities on XE520 processors.

Archive. To archive is to back up system files, usually onto tape.

Archive Volume. An archive volume is a disk or (most often) a tape used to archive (back up) files.

ASCII (American National Standard Code for Information Interchange). ASCII is a set of 7–bit coded characters (8 bits including parity check) used for information interchange between data communications systems.

Background Process. A background process is a process that, once begun, runs with no user interaction through the terminal. Another process is then free to use the terminal.

Back Up. See Archive.

Bad Spot. A bad spot is a faulty part of a disk, where information cannot be stored and read reliably.

Block. A block is a 512–byte subdivision of data on a disk. Blocks are also referred to as sectors.

Bootup. Bootup is the operation that starts up the system when the keyswitch on the XE520 base enclosure is turned from STOP to MANUAL, REMOTE, or NORMAL. The bootup operation includes running a series of self–diagnostic tests, loading the master operating system, and reading the set of system files associated with the position the keyswitch is turned to.

BTOS. BTOS is any Unisys operating system. Different XE520 processor boards run different versions of a system called XEBTOS.

BTOS Executive. See Executive.

BTOS Status Codes. BTOS status codes are numerical codes reported by the system software through the BTOS Executive and other facilities. These codes indicate the status of system operations.

BTOS Workstation. A BTOS workstation is a Unisys model B26, B27, B28, or B38 workstation.

Bucket. When the mHistogrammer utility runs, the address range to be monitored for CPU activity is divided into areas called buckets.

Built–In Disk Device. A built–in disk device is a 5 1/4–inch hard disk drive. An SMD disk device is not considered a built–in, even if housed in an XE520 base enclosure.

Central Processing Unit (CPU). In a computer, the unit that performs most of the fundamental processing is the central processing unit.

CLI. See Command Line Interpreter.

Cluster Line. A cluster line is an RS–422 cluster communications port connection that allows workstations to communicate with the XE520.

Cluster Processor (CP). A cluster processor (CP) is an XE520 processing board that runs communications software and supports workstations, a parallel printer, and up to three RS-232-C serial devices.

Command. A BTOS command is an instruction, entered from the BTOS Executive, that causes a corresponding utility to run on a BTOS processor. Each command has a command name, and most have command forms.

Command File. A command file is created by the system administrator and contains a listing of the BTOS commands available to a specific user.

Command Form. A command form is an interactive form, associated with many BTOS commands, where a user specifies parameters to define how the command is to be executed.

Command Line Interpreter (CLI). The Command Line Interpreter is an interface program that interprets statements and executes BTOS run files on XE520 processors.

CP. See Cluster Processor.

CPU. See Central Processing Unit.

Crash Dump. A crash dump is copy of a processor's memory image at the time of a nonrecoverable failure (crash). Unisys personnel can use it to analyze the cause of a processor failure.

Customized Mode. When booted up with a customized set of system configuration files, a system is said to be operating in customized mode.

Cylinder. A cylinder is a set of disk tracks that can be accessed as a unit.

DAM. See Direct Access Method.

Data Volume. A data volume is a volume used for data and program storage.

Debugger. The Debugger is a program that detects errors (bugs) in programs. It allows you to examine and modify memory, to set and clear breakpoints, and to produce formatted displays of memory.

Demand Paging. Demand paging is a form of memory management that keeps in onboard memory only those parts of the program code and data required for execution.

Device. A device is a piece of hardware, such as a terminal, printer, tape drive, or other input/output unit.

Direct Access Method (DAM). The direct access method is a file access method providing random and non–overlapped input or output of records.

Directory. A directory is a group of related files. Volumes contain directories, and directories contain files, in the BTOS three–level hierarchical file system.

Disk Device. A disk device is a hardware component that stores data. XE520 disk devices are built–in or Storage Module Device (SMD).

Disk Extent. A disk extent is a set of one or more contiguous disk sectors that contain all or part of a file.

Disk Processor (DP). A disk processor is an XE520 processor board consisting of a storage processor and a storage contoller. The DP supports I/O to half–inch magnetic tape drives and SMD disks.

DP. See Disk Processor.

ECC. See Error Correction Code.

Error Correction Code (ECC). Error Correction Code is a method of detecting and correcting data errors with a four–byte block check character appended to the data block.

Executive. The Executive is an interactive BTOS application program that allows you to execute various BTOS utilities through command forms.

File Processor (FP). A file processor (FP) is an XE520 processor board that supports I/O operations to disk devices.

FP. See File Processor.

File System. A file system is a collection of files stored on the same logical disk device. A BTOS file system is made up of a three–level hierarchy that includes a volume, its directories, and their files.

Hexadecimal. Hexadecimal is a system of representing numbers in base 16.

Home Directory. The home directory is the directory a user automatically enters upon logging on to the system.

ICC. See Inter–CPU Communication.

Installation Log. An installation log is a log that provides a record of software installation operations.

Inter–CPU Communication (ICC). Inter–CPU communication is the protocol used by XE520 processors when communicating across the system bus.

I/O. I/O stands for Input/Output.

I/O Device Configuration File. An I/O device configuration file defines the hardware and software parameters of disk drives, printers, tape drives, and communications devices such as modems and RS–232–C serial terminals.

JCL. See Job Control Language.

Job Control Language (JCL). Job Control Language is the language in which statements are submitted to CLI for batch execution.

KB. See Kilobyte.

Kilobyte. One kilobyte is 1024 bytes.

Master Commands. Master commands are BTOS commands, executed from the Executive, that invoke utilities specific to the XE520.

Master Configuration File. The master processor (FP00 or DP00) uses the master configuration file to determine the processor operating systems to be loaded at boot time.

MB. See Megabyte.

mCommands. See Master Commands.

MD3. See Multidisk 3.

ME. See Memory Expansion Board.

Megabyte. One megabyte is 1,024,000 bytes.

Memory Expansion (ME) Board. A memory expansion board attaches to a processor board to supply additional processor memory capacity.

Multidisk 3 (MD3). A Multidisk 3 is a freestanding enclosure supporting up to three 135–MB (formatted) SMD disk drives.

Normal Mode. When the keyswitch is turned to NORMAL and the system is booted up with the default set of system files, the system is said to be operating in normal mode.

Null String. A null string is a sequence of blank characters. It usually does not change the default condition set up in the original BTOS command or submit file.

Operating Mode. The XE520's operating mode is determined by any one of four sets of system configuration files and BTOS processor versions used to boot up the XE520. The particular set of system configuration files used during boot–up time depends on the keyswitch position.

Operating System. An operating system is the part of the system software that supervises the running of individual programs. It loads programs, allows concurrent operation of two or more programs, schedules processes within the system, and manages information. The XE520's operating system is XEBTOS. The operating system is sometimes referred to as the system image.

Partition. A partition is an assigned area of processor memory. A program the processor is running can be assigned a partition; the program restricts its instruction and data to this area. The processor can then operate in a multitasking mode, with several programs sharing processing time. See Primary Partition and Secondary Partition.

Password. A password is a character string used as part of the BTOS file security system. Users, devices, volumes, directories, and files can have passwords. In conjunction with protection levels, passwords determine access to information.

Path Name. A path name is the description, including volume and directory names, of a file's position in the file system.

Port. A port is a part of a processor dedicated to a single data channel for receiving data from, or transmitting data to, one or more external devices.

Primary Partition. A primary partition is an assigned area of an XE520 processor's memory that is established during system start–up. Services that require user interaction and some system services must run in the primary partition. Each processor has only one primary partition. See Partition.

Printer Spooler. The printer spooler is a system service that manages the transfer of data from disk files to printers.

Processor Configuration File. The processor configuration file defines the hardware and software configuration associated with a processor board.

Processor Initialization File. The processor initialization file defines the system services to be run on a processor when the system is started up.

Product Content File. The product content file contains entries for each new software product stored on a release disk. This file provides the Boot Load utility with instructions for loading the products.

Protection Level. A protection level is a decimal value assigned to a file that determines the type of password, if any, that a user must enter to gain read or write access to that file.

QIC Tape. See Quarter-Inch Cartridge tape.

Quarter–Inch Cartridge (QIC) Tape. Quarter–Inch Cartridge (QIC) tape is quarter–inch–wide magnetic storage tape contained in a hard plastic cartridge.

Queue Manager. Queue is a BTOS system service that controls the various queues in which tasks are stored for processing. For example, it controls the queues that handle print requests made through printer spoolers. It can also manage similar queues for data communications processes and user-defined processes.

RAM. See Random-Access Memory.

Random-Access Memory (RAM). Random-access memory (RAM) is the processor memory to which and from which data is written and read nonsequentially.

Read-Only Memory (ROM). Read-only memory (ROM) is the processor memory whose contents can be read but can be written only by special means. The ROM normally contains start-up programs, such as self-diagnostic tests.

Register. A register is a temporary memory location.

Restore. To restore is to rewrite archive files to an XE520 volume.

Restricted Mode. When the XE520 system starts up with the keyswitch set to REMOTE and reads the set of system configuration files that contain the identifying character .r, it is said to be operating in restricted mode.

ROM. See Read-Only Memory.

RS-232-C. RS-232-C is an industry specification developed to standardize the interface between different types of communications equipment.

RS-422. RS-422 is a high-speed communications standard used in an XE520 system to link cluster workstations.

Sector. A sector is the smallest addressable portion of a track or band on a hard or floppy disk. (See Block.)

Secondary Partition. A secondary partition is an assigned area of processor memory where applications that do not require user interaction, including some system services, run.

Sign-On File. A sign-on file is required for each user to log into the system. Among other things, this file defines the user's password, the BTOS commands the user can access, the path (volume and directory) where the user will be placed, and the application the user will enter.

SMD. See Storage Module Device.

SP. See Storage Processor.

Spooler. A spooler is a BTOS utility that manages the operation of printers. A spooler uses special files called queues to store print requests until a printer becomes available.

Spooler Configuration File. A spooler configuration file contains information the system uses to coordinate the spooler and the queue manager.

Status Codes. See BTOS Status Codes.

STATUS Display. XE520s report the status of software installation, system start–up operations, and error conditions through STATUS displays on their base enclosures.

Storage Controller (SC). A storage controller is a board that combines with a storage processor to form a disk processor and that controls SMDs.

Storage Module Device (SMD). A storage module device is a 133–byte (formatted) Memorex 166 disk drive.

Storage Processor (SP). A storage processor (SP) is an XE520 processor board that controls half–inch magnetic tape.

Subcommand. A subcommand appears within the operational procedures of a command and makes an additional operation available.

Surface Test. A surface test is a disk initialization operation that checks for bad spots on a disk by writing data to and then reading the data from each sector of the disk.

System Bus. The system bus is the hardware device over which the system processors communicate.

System Configuration Files. System configuration files are text files that define hardware and software configuration.

System Crash Status Words. System crash status words normally display at the terminal screen and are entered in the system log after a system crash occurs. They are useful for isolating the cause of a system crash.

System Image. A system image is the run file for an operating system.

System Log. The system log is displayable through the MPLOG command and contains a detailed account of system status, errors, and failures.

System Volume. A system volume is a volume you can boot the XE520 from.

Terminal. A terminal is a device, such as a workstation, usually equipped with a keyboard and a display screen and capable of sending and receiving information over a communication channel.

Terminal Processor (TP). A terminal processor (TP) is an XE520 processor board that supports a parallel printer and up to ten RS–232–C serial devices.

TP. See Terminal Processor.

Utility. A BTOS utility is a program that is invoked by executing a command from the Executive or by executing a utility's run file through the Command Line Interpreter (CLI) using Job Control Language (JCL).

VHB. See Volume Home Block.

Volume. In BTOS, the complete file system unit of information stored on a formatted disk is a volume.

Volume Fragmentation. Volume fragmentation is the scattering of disk extents available for data storage. A volume is said to be fragmented if the system must allocate two or more noncontiguous disk extents to fulfill a data storage request. See Disk Extent.

Volume Home Block (VHB). A volume home block is an area on a volume that contains the volume name, the date it was created, the date it was last modified, and the number of free pages and file headers.

Volume Initialization. Volume initialization is the procedure by which the mlVolume utility prepares an XE520 disk for use as a system volume. Among other tasks, this procedure formats the disk surfaces, performs read/write tests, writes volume control structures, and creates system files.

X, Y, Z Blocks. X, Y, and Z blocks are operating system parameters that control the number and size of cluster communication blocks.

Wild Card Character. The BTOS system allows two wild card characters to be used in command form fields: an asterisk (*) and a question mark (?). The asterisk represents any string of characters; the question mark represents any individual character. Some operations allow you to use wild card characters in file specifications. The system then tries to match the portion of the name that appears before or after the wild card character and performs the requested operation for each matched file name.

Index

C

D

E

P

Q

R

S

T

U

V

W

X

Help Us To Help You

Publication Title

Form Number Date

Unisys Corporation is interested in your comments and suggestions regarding this manual. We will use them to improve the quality of your Product Information. Please check type of suggestion:

☐ Addition ☐ Deletion ☐ Revision ☐ Error

Comments

Name

Title Company

Address (Street, City, State, Zip)

Telephone Number

Help Us To Help You

Publication Title

Form Number Date

Unisys Corporation is interested in your comments and suggestions regarding this manual. We will use them to improve the quality of your Product Information. Please check type of suggestion:

☐ Addition ☐ Deletion ☐ Revision ☐ Error

Comments

Name

Title Company

Address (Street, City, State, Zip)

Telephone Number

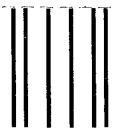
No Postage
necessary
if mailed in the
United States

BUSINESS REPLY MAIL

First Class Permit No. 817 Detroit, MI 48232

Postage Will Be Paid By Addressee

Unisys Corporation
ATTN: Corporate Product Information
P.O. Box 418
Detroit, MI 48232-9975 USA



No Postage
necessary
if mailed in the
United States

BUSINESS REPLY MAIL

First Class Permit No. 817 Detroit, MI 48232

Postage Will Be Paid By Addressee

Unisys Corporation
ATTN: Corporate Product Information
P.O. Box 418
Detroit, MI 48232-9975 USA